Spring 5-21-2018

Clock Jitter in Communication Systems

Andrew Wayne Martwick
Portland State University

Clock Jitter in Communication Systems

by

Andrew Wayne Martwick

A dissertation submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy
in
Applied Physics

Dissertation Committee:
Raj Solanki, Chair
Andrew Rice
Jim O'Donnell
Gary Goncher
James Morris

Portland State University
2018

ABSTRACT

For reliable digital communication between devices, the sources that contribute to data sampling errors must be properly modeled and understood. Clock jitter is one such error source occurring during data transfer between integrated circuits. Clock jitter is a noise source in a communication link similar to electrical noise, but is a time domain noise variable affecting many different parts of the sampling process. Presented in this dissertation, the clock jitter effect on sampling is modeled for communication systems with the degree of accuracy needed for modern high speed data communication. The models developed and presented here have been used to develop the clocking specifications and silicon budgets for industry standards such as PCI Express, USB3.0, GDDR5 Memory, and HBM Memory interfaces.

ACKNOWLEDGEMENTS

My thanks and best wishes goes to my Portland State advisors and friends, Dr. Solanki, Dr. Leung, and Dr. Rice, for their years of support and encouragement in my educational endeavors. This work has been made possible by the support of my management, my colleagues, and friends at Intel Corporation over the last 22 years. In particular, I owe a debt of gratitude to John Grooms, Henry Peng, Dan Plitt, Duane Quiet, Kevin Slattery, Ron Swartz, Mike Li, and Randy Wilhelm for their generous support of a very unconventional career path.

None of this work would have been possible without the encouragement, hard work, and sacrifice by my loving wife, Alica, and my very supportive family.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction and Background Information

1.1 Introduction

Modern electrical communication systems send bits of data from one component to another in a reliable, energy efficient method. Bits of data can be sent over wires of a cable, traces of a printed circuit board, as waves through space, or as light on a fiber optic cable. Transmitted bits are sent at time intervals mT are sampled at time intervals nT by the receiving device, where T is the ideal sample period and m,n are integers. The receiver voltage sampling function returns a value of 0 or 1 based on a voltage comparison with a threshold voltage,

1-1
$$v(nT) = \begin{cases} 1, & v(nT) \geq V_t + V_{noise} \\ 0, & v(nT) < V_t - V_{noise} \end{cases}$$

where V_noise is one half the total noise present at the amplifier. In order to be properly sampled, the time of the sampling must occur when the voltage is beyond the noise threshold or the resulting sample might be incorrect.

Like all communication, the sending of messages occurs in the presence of many types of environmental noise. Thermal noise at an amplifier is an example of an electrical noise that manifests itself as static on a radio. This noise interferes with the sampling of the bit because it adds uncertainty to the threshold voltage as shown in Equation 1-1. Electrical noise interference is well understood as a signal to noise ratio, and the work of Shannon [1] developed expressions for quantifying the channel capacity of a

communication system given knowledge about the amplitude of a signal in the presence of electrical noise. The study being undertaken here presents the error contribution in the sampling process due to the clock jitter present in the system, an error in the sampling process due to shifts in time. The clock jitter error is a contribution to the probability that the data bit will be sampled incorrectly, thus causing a bit error. By calculating the probability of a sampling error due to the clock jitter and all of the other error sources, the bit error rate can be estimated and the appropriate error codes can be designed to protect the data integrity.

Communication standards exist to ensure different components can be designed by different entities and will interoperate with each other. The standards allocate the bit time error to different processes for the link. Communications standards are now in development with bit times of 31.25 picoseconds, and research is underway for bit times of twenty picosecond bit times and less. At these data rates, the timing budget requires sub picosecond accuracy in the in order to have interoperable components. Jitter processes that were previously ignored must now be modeled and the contribution must be accounted for in order to ensure the error detection and correction codes designed for these links are sufficient for reliable data transmission.

1.2 Contribution of the Dissertation

The models developed in this work have been used to develop the timing budgets for industry communication interface standards such as PCI Express (PCIe), High Bandwidth Memory (HBM), GDDR5, and USB3.0. As the data rates have increased, the timing budgets have gone from 100s of picoseconds to currently budgeting femtoseconds

of timing allocation. This has required continuous improvement of these models to increase accuracy and account for effects that could previously be ignored. Refinements presented in this work include the non-linear behavior of power delivery networks, digital loop tracking error, and electrical noise fluctuations on the clock due to crosstalk or other electrical sources. This work will enable these higher speed standards to achieve the goal of component interoperability.

1.3 Jitter Definitions

For the purposes of jitter modeling, the center of the data bit is the ideal time to sample the data bit. Jitter is then the time difference between the center of the data bit when the sampling should occur to the time sampling actually occurs. The sampling clock and the data are both periodic functions, thus jitter is a sampled variable that follows the known discrete mathematics of sampling theory. The clock ideally occurs at times nT, where n is an integer. If the clock moves farther than one half of the bit period, a sampling error will occur, and this is considered to be a bit error.

Some types of specifications, like Ethernet, define the data relative to an ideal clock. This makes the jitter defined relative to an absolute time and the system is architected to maintain an accuracy to absolute time. Other architectures define the jitter relative to the data, since the clock – data alignment is what determines if the sampling operation is a success or results in a bit error. The advantage of a relative timing architecture is the jitter rejection when the jitter is correlated on both the data and the clock. Correlated jitter on both the clock and data is called common mode jitter.

1.3.1 Phase Jitter

Phase is a measure of location within the cycle time of a periodic function. Phase equal to zero corresponds to the beginning of the bit, and a phase of 360 degrees or 1 radian (2 pi) corresponds to the end of the bit. Absolute phase is a variable that is 0 at t=0 and increments in time at the rate n*T, where n is the bit number. Absolute phase for an ideal clock and a clock with jitter is shown in Figure 1.

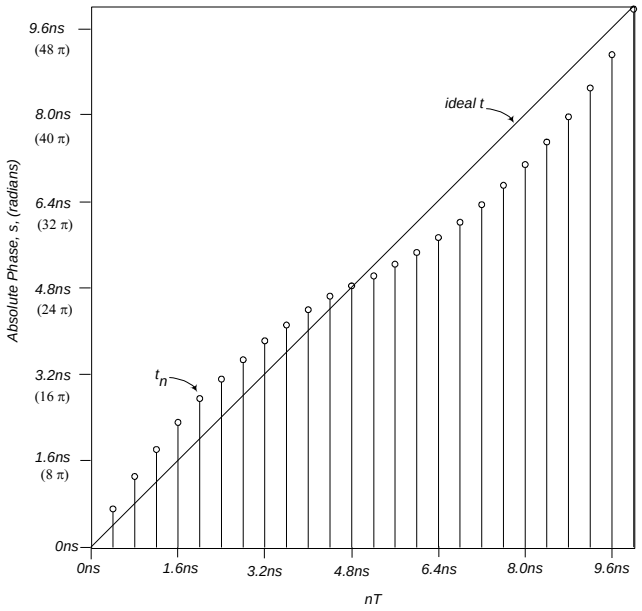

Figure 1: Absolute Phase and the Ideal Clock

An ideal sampling clock would measure the data at the time nT, as shown above. The actual clock will sample the data at the time t_n, that can be different than the ideal time. The jitter is the difference between when the crossing actually occurred, tn, and when it should have occurred with an ideal clock, nT. This difference is the phase jitter,

1-2 $$\Phi_n = t_n - nT,$$

and is shown in Figure 2.

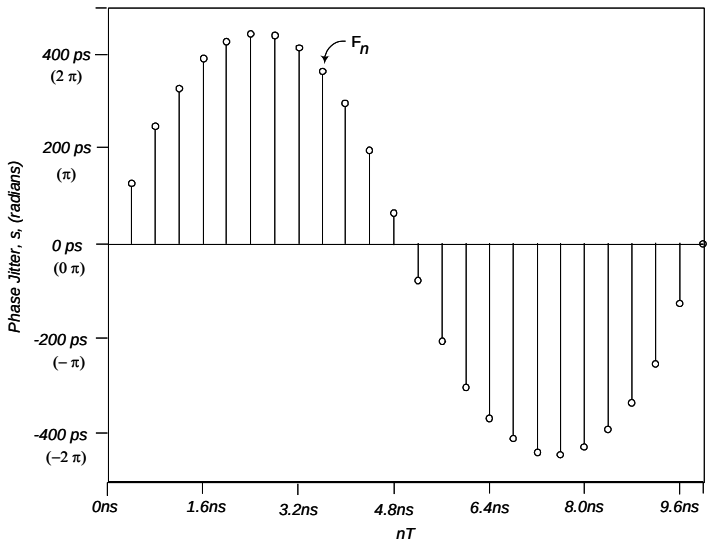

Figure 2: Phase Jitter after Clock Recovery

1.3.2 Period Jitter

The period Jitter (Φ') is the difference between the measured period and the ideal period and is

1-3 $$\Phi'_n = (t_n - t_{n-1}) - T.$$

The period jitter is also the first difference of the phase jitter,

1-4 $$\Phi'_n = \Phi_n - \Phi_{n-1}.$$

If the phase jitter is known, the first difference gives the period jitter. Conversely, if the period jitter is known the phase jitter can be obtained by taking the cumulative sum of the period jitter.

An example of period jitter, Φ', is shown in Figure 3. The Y axis shows the magnitude of the period jitter in ps and, parenthetically, in radians.

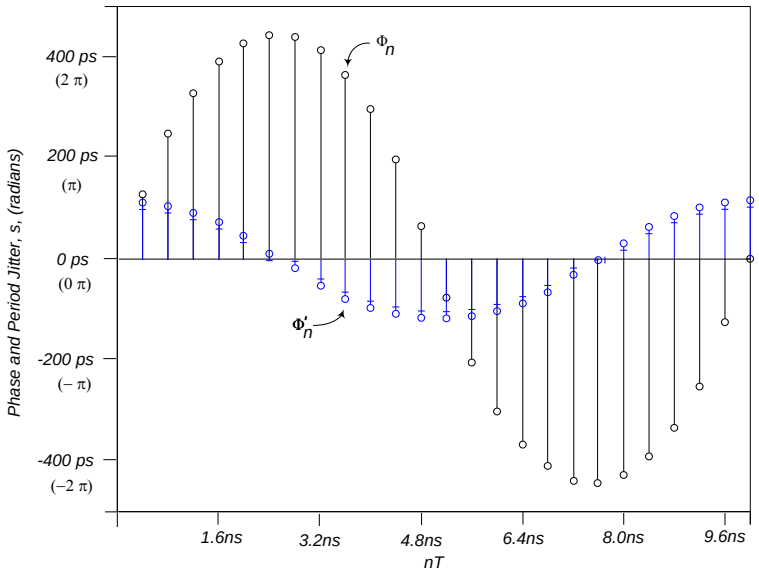

Figure 3: Phase and Period Jitter vs nT

1.3.3 Cycle to Cycle Jitter

The second difference of the phase jitter gives the cycle to cycle jitter, the difference between two consecutive cycles. This is

1-5 $$\Phi''_n = (t_n - t_{n-1}) - (t_{n-1} - t_{n-2}) .$$

The expression on the right hand side is the difference between two adjacent periods and is independent of the clock jitter.

1.3.4 Frequency Response of Jitter

The frequency response of the different jitter representations is found by substituting $e^{j\omega t}$ for t_n and finding the magnitude of the response. It can be shown that this is given by the frequency response of a difference function,

1-6 $$H(f) = 2\sin\left(\frac{f\pi}{f_s}\right),$$

where f_s is the sampling frequency. This frequency response is shown in Figure 4, where the frequency axis is normalized by the sampling frequency.

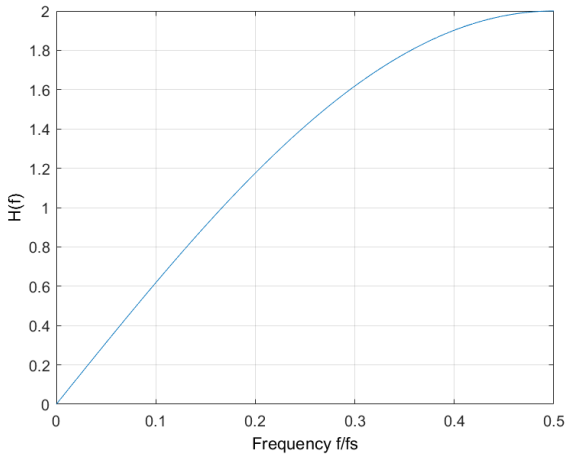

Figure 4: Normalized Frequency Response of Phase / Period Jitter

This can be repeated for higher order differences, for the nth order, as in

1-7 $$H^{(n)}(f) = \left[2sin\left(\pi \frac{f_{jitter}}{f_s}\right)\right]^n.$$

The case of n=2 is shown below in Figure 5.

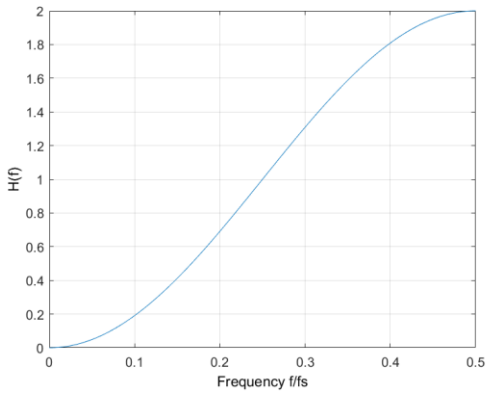

Figure 5: Normalized Frequency Response of Phase / Cycle-Cycle Jitter

This shows how the difference operation on a vector amplifies frequencies greater than 0.18 times the Nyquist frequency and attenuates frequencies below this point. Thus, reporting the jitter magnitudes as phase jitter, period jitter or cycle to cycle jitter provides the same information up to a constant of integration, with different frequency components attenuated or amplified.

1.3.5 N Cycle Jitter

Some specifications have included a term called "N-Cycle" jitter that is intended to constrain the jitter over multiple cycles. N cycle jitter is yet another way of looking at the jitter through a frequency filter response.

Let $J_n(N)$ be the jitter at sample n. Defining the N cycle between sample n-N/2 and n+N/2, we have the following definition of N cycle jitter:

1-8 $$J_{n(N)} = t_{n+\frac{N}{2}} - t_{n-\frac{N}{2}}$$

The frequency response can be found by substituting $t_{m=}e^{j\omega nT}$ and solving the eigenvector equation, giving

1-9 $$H(f) = |i2\sin(\pi NTf)|$$

Letting fT = 0.5 (the Nyquist frequency), the frequency response of N cycle jitter is shown for three different values of N in Figure 6.

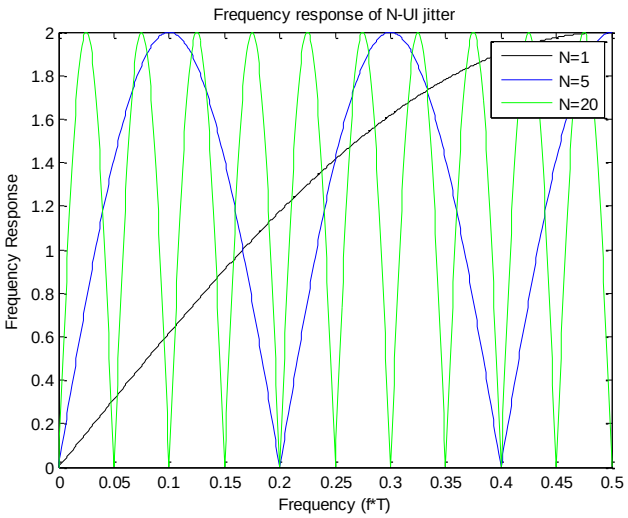

Figure 6: Frequency Response of N-Cycle Jitter Measurement

As shown above, N cycle jitter acts as a comb filter on the jitter measurement. This type of measurement only makes sense if there is sampling in the circuits of data sampled by a clock at some NT periods later. As an example of such a circuit, consider the clock delayed by N cycles due to a clock tree distribution. This is common on a parallel bus such as the JEDEC standard Double Data Rate (DDR) memory interface. For this standard he jitter transfer function magnitude is a comb filter.

1.4 Jitter Relative to the Recovered Clock

For many serial links the clock is recovered from the data. The clock recovery function is defined by the serial specification and follows the data, attempting to place the clock accurately relative to the data. The absolute phase of the recovered clock follows the absolute phase of the data in a way specified by the clock recovery function. This is shown in Figure 7. The data is received with jitter, and the recovered clock, warped t, is generated in a way that tracks the data as accurately as possible. Because of the time

required to process the tracking loop, the recovered clock will never track the data exactly, resulting in a sampling time that is less than the ideal position.

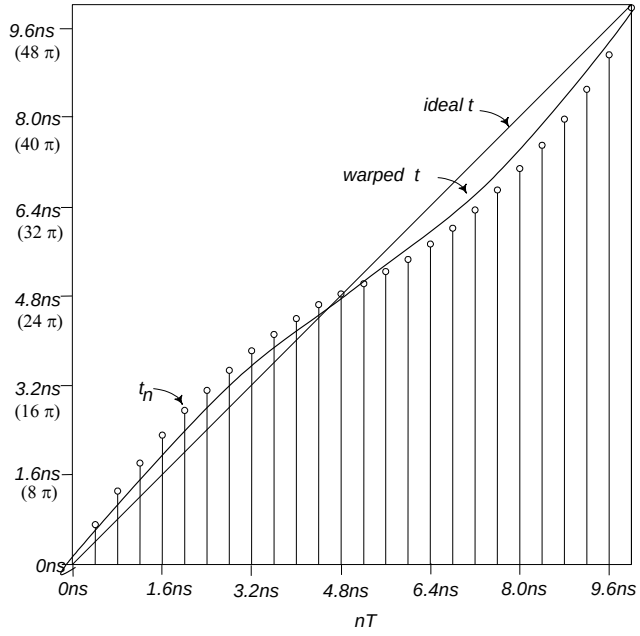

Figure 7: Clock Recovery with Ideal T and Recovered T

The clock recovery functions are high pass functions since this properly models the way the CDR circuit behaves. Later sections discuss the recovered clock and tracking functions in more detail.

1.5 Spread Spectrum Clocks

Spread spectrum clocks (SSC) are clocks that have added a large, low frequency jitter source that is used to lower electromagnetic emission measurement at any one particular frequency for FCC EMI compliance purposes. The standard SSC phase jitter component is due to a 0.5% frequency reduction that oscillates at 31.25 KHz. This means

the clock frequency is changed between f to 0.0995 f at a rate of 31.25 KHz. The average frequency is

$$f_{avg} = \frac{f + 0.995\, f}{2} \quad \text{1-10}$$

.

The maximum phase deviation from the ideal can be calculated by integrating the instantaneous frequency and subtracting out the average for all the times the UI is over (or under) the average UI. This amounts to calculating the shaded area in Figure 8 with f_average = 0. There are other spread spectrum profiles that are shaped differently; margin is required for those profiles.

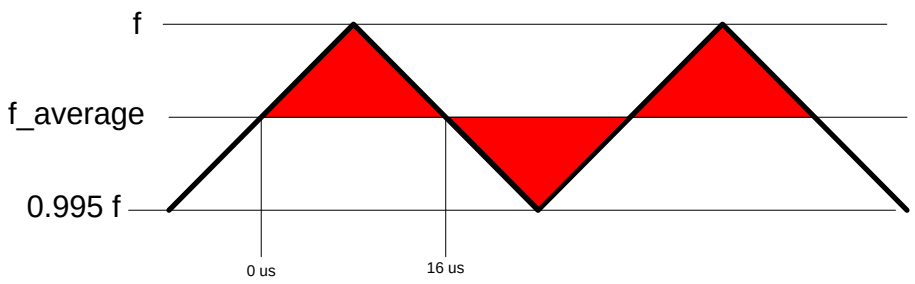

Figure 8: SSC Frequency Modulation Profile

The shaded area in Figure 8 is ½ the base times the height, giving the phase deviation of

$$\phi_{\max} = 20\, ns \quad f \quad \text{1-11}$$

,

in units of radians. The phase starts from 0 and increases to 20 ns, then decreases back to 0. This can be approximated by a sinusoid of amplitude 10 ns.

The SSC period jitter is obtained using $\Phi'_n = (t_n - t_{n-1}) - T$ and the definition $\Delta T = t_n - t_{n-1}$. Solving for the maximum period difference,

$$\Delta T = \frac{1}{f - 0.005 f} \sim 1.005T \tag{1-12}$$

the maximum period jitter is $\Phi'_{max} \sim 0.005\,T$. Graphically, the period jitter is shown in Figure 9.

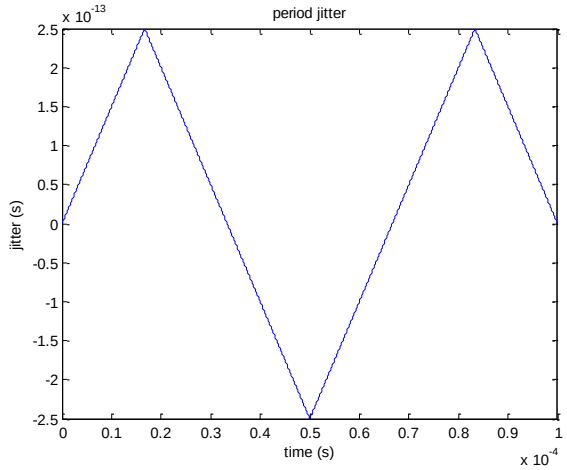

Figure 9: Period Jitter of SSC

Taking the cumulative sum of the period jitter gives the phase jitter. This is shown in Figure 10 along with the sinusoid approximation.

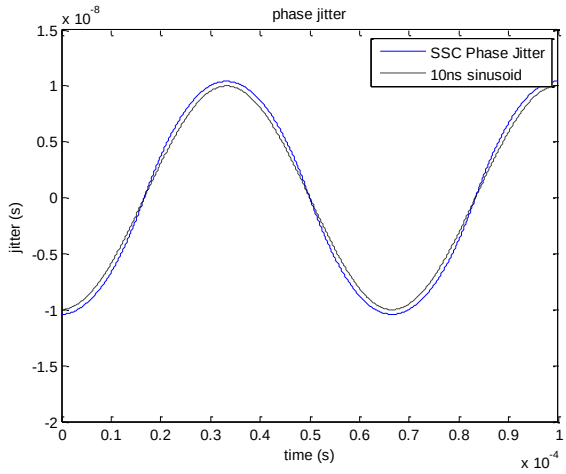

Figure 10: Phase Jitter of SSC

In some communication architectures, there is independent SSC on the transmitter and the receiver. This effectively doubles the amplitude of the SSC phase jitter that must be tracked as a nearly sinusoidal jitter with an amplitude of 20 ns (40 ns peak to peak) With these assumptions the maximum slew rate that must be tracked is

$$\frac{d}{dt}\left(40x10^{-9}(2*pi*33{,}000*t)\right)_{\max}=8.3\,\frac{m\sec}{\sec}\quad . \tag{1-13}$$

It will be shown in a following section on the CDR, this is a limiting requirement on the actual CDR and results in CDR tracking error in the form of dither. This will be described in detail in a later chapter.

1.6 Communication Systems

Digital data is transmitted from the transmitter to the receiver, where it is sampled by a circuit at a time dictated by a clock, and a 1 or a 0 is registered. This is true for all communication systems regardless of the architecture details. In this section the

important circuit blocks are described and some of the common communication architectures are introduced.

1.6.1 Receiver Sampling

The receiver is normally implemented as a differential flip flop. An example of a typical Strong Arm latch [4] receiver circuit is shown in Figure 11.

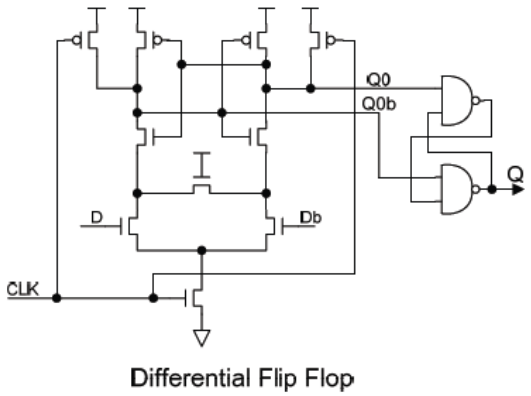

Figure 11: A Strong Arm Sampling Latch

This receiver needs sufficient voltage margin between DQ and Vref (labeled D and Db) to exceed the uncertainties in the offset of the voltages and the common mode noise injected into the circuit by the DQ line by the sampling clock. This type of budget, due to manufacturing and fixed offsets, is a deterministic uncertainty. The minimum input voltage required, Vmin, and is shown in relation to the eye diagram in Figure 12.

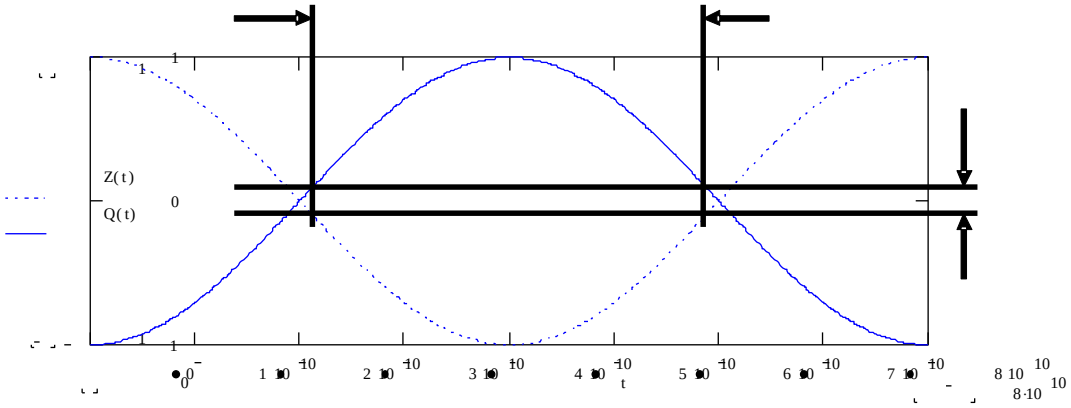

Figure 12: Eye Diagram for a Strong Arm Latch

The sampling process itself takes only a few ps due to the positive feedback in the sampling latch. For jitter calculation purposes it is reasonable to assume the sampling occurs at a particular voltage and occurs instantaneously. An improvement is to include the integration effect of the input capacitance and finite channel impedance on the signal, this is in effect a low pass filter on the signal. Typical low pass filter values for a high speed signal are 0.2 pf * 50 ohms, or on the order of 15 GHz.

1.6.2 Phase Lock Loops

A phase lock loop (PLL) consists of a controllable ring oscillator with a phase that is adjusted to match a reference [5]. A PLL block diagram showing the major functions is Figure 13.

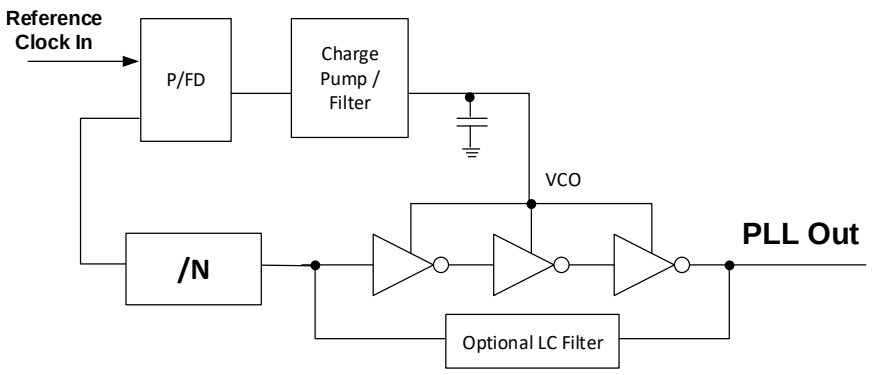

Figure 13: PLL Block Diagram

The voltage controlled ring oscillator (VCO) is designed to run at a preferred frequency by tuning the delay of the inverter stages and optionally including a LC filter. Ring oscillators of 10 GHz are common. The current ring oscillator operating frequency is generally limited by thermal considerations and not by the process technology.

Phase / Frequency detector compares the input data phase to the output of a voltage controlled oscillator at every reference clock transition. By dividing the VCO by N, the PLL output frequency can be much higher than the reference clock. If the phase of the reference clock occurs before the VCO clock, the reference clock is early. The PFD tells the charge pump to increase the frequency of the VCO by raising the voltage of the VCO proportional to the phase error. This increases the frequency of the VCO and makes the VCO phase better align with the data at the next reference clock. Conversely, if the VCO phase is early, the VCO voltage is lowered. These outputs are filtered in order to filter input jitter on the reference clock.

There are many modern digital innovations in current PLL design. The PFD can be replaced with a high speed thermometer encoded phase comparator, the tank oscillator

can be implemented with digitally tunable capacitors to adjust the natural frequency, and it is now common to implement the charge pump filter and analog filter as a digital IIR filter with a D/A converter output. Regardless of the implementation, the transfer function of the PLL is defined as how changes in the input phase of the reference transfer into changes of the output phase of the VCO and will be in later sections.

1.6.3 Delay Lock Loop

A DLL is very similar to a PLL with the exception of the VCO. The delay lock loop can be used to divide a clock into equal phases, based on the number of inverters used and the feedback insertion delay. In a DLL, the voltage on an inverter delay chain is adjusted to adjust the propagation delay through the chain, ideally with equal phase spacing through each inverter. Unlike a PLL, the DLL cannot multiply the frequency. A phase interpolating DLL used in clock and data recovery (CDR) is described later in greater detail. A DLL block diagram is shown in Figure 14.

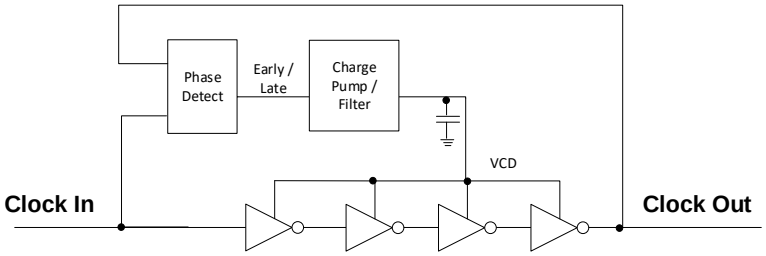

Figure 14: DLL Block Diagram

1.6.4 Parallel Strobe / Source Synchronous Architectures

One of the simplest implementations of a communication system is a parallel bus, where the source synchronous clock is sent along with the N bits of data. A source synchronous link is shown in Figure 15.

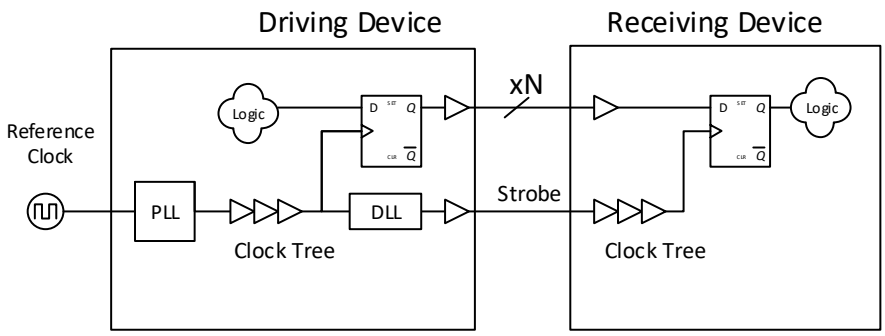

Figure 15: Overview of a Clock Forwarded System

In this example, the reference clock is multiplied to the data rate frequency by the PLL and distributed to the I/O blocks through a clock tree. The clock is used to present the N bits of data onto the physical channels and another copy of the clock is phase adjusted by the delay lock loop (DLL) and output as a strobe. The strobe is sent to the receiving device at the data rate. The phase is normally adjusted by the DLL during a training phase to clock the average center of the received data bit using a loopback mechanism in the logic. The clock jitter at the sampling latch is the total clock jitter that is not common with the data jitter.

1.6.5 Mesochronous Serial Architecture

An implementation that recovers the clock from the data is shown in Figure 16. This implementation is called a mesochronous system as there is only one reference clock for all the components. For this architecture the instantaneous phase may be different at different parts of the circuit but the frequency is the same everywhere, on average. The PLLs multiply the reference clock frequency to the data rate as needed. The difference in phase of any of the clocks at any point in the system can vary due to time delay

differences and different noise sources. The phase difference is dynamic and can vary more than several bits. It is the function of the clock recovery block to adjust for the several bit time variance of the phase between the clock and data.

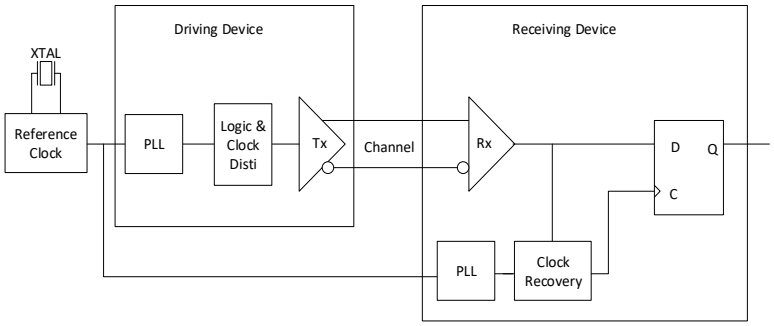

Figure 16: Overview of a Mesochronous System

In this example, the data is sent on the channel using a clock source to time the bit transmissions. It is sampled by the receiver clock that is recovered using both the reference clock and the data itself. This receiver sampling clock being generated by the clock recovery block is an estimate of the ideal clock, identical to the “warped T” or ideal T previously given in the definition of phase hitter. The reference clock is normally a low frequency and multiplied to the data rate by a PLL. Reference clock frequencies of 25 MHz – 156 MHz are common.

1.6.6 Plesochronous Serial Architecture

A plesochronous system has two separate clocks that are almost, but not exactly, the same frequency. This is typical for Ethernet and is show in Figure 17.

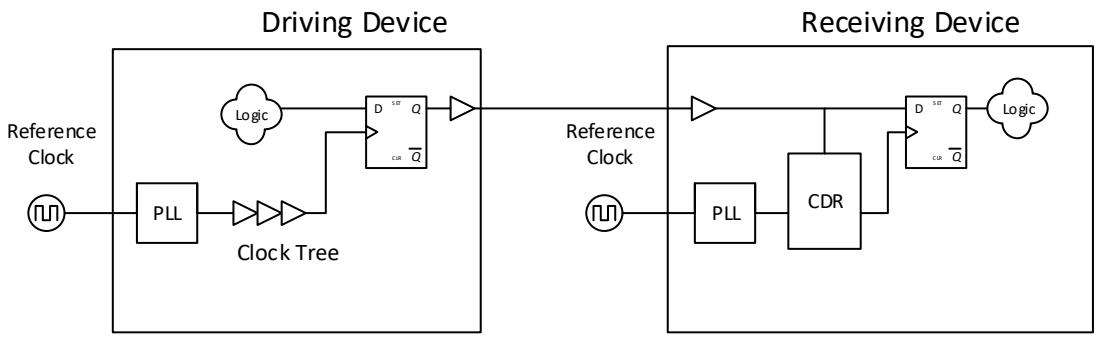

Figure 17: Overview of a Plesochronous System

In this system, the clock phase is continually adjusted by the receiver CDR to try to stay in the center of the data bit. There is phase noise on the data that originates in the reference clock, the clock tree, and other circuits, and there is phase noise on the CDR that generates the sampling clock. The phase difference between the driving device and the receiving device can drift any amount over time. In some architectures, the reference clocks contain spread spectrum jitter components. The CDR can operate by either looking exclusively at the data or by a combination of using the reference clock in conjunction with the recovered data phase.

1.7 Jitter Statistics

Many of the noise processes in a communication link are stochastic, that is, they vary randomly due to fluctuations of electrons or due to the random nature of the data being sent. This section provides a background on the statistical nature of bit errors caused by jitter. By applying statistical methods, much higher data rates are possible than using absolutely bounded noise budgets. This method, however, leaves open the

possibility that the transfer will be an error because the jitter process can push the sampling clock into sampling the wrong bit, or sampling when the voltage is not sufficient to drive the sampling latch.

1.7.1 Bit Error Rate

High speed interfaces are often defined using an eye diagram. The eye diagram presents both the clock recovery function and the statistical nature of the high speed link that is critical to budgeting both timing and voltage. An example of an eye diagram is shown in Figure 18.

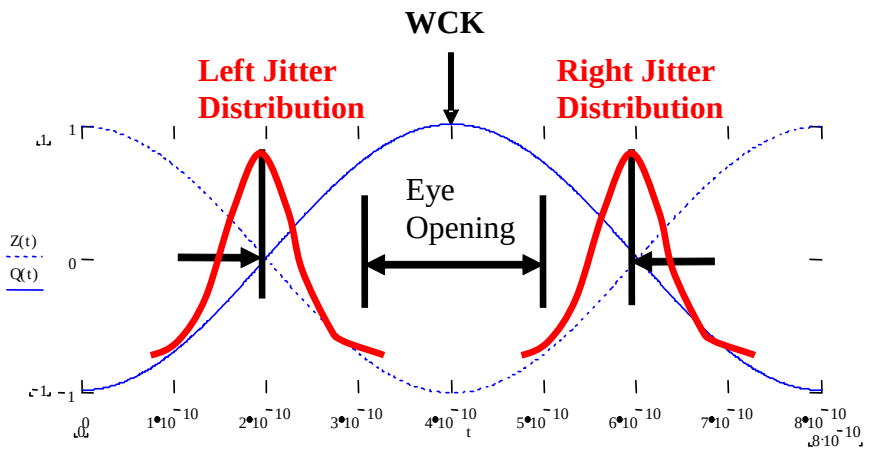

Figure 18: Example Eye Diagram with Jitter Distribution

In this figure, the center of the diagram locates the position of the sample clock while the distribution about the edges indicates the statistical nature of jitter relative to the sampling clock. The edges are the transition regions of the data lines (DQ) being

sampled by the sampling clock, WCK. An error occurs if the DQ crosses on the wrong side of WCK. This is the common representation since it matches measurements taken with an oscilloscope, where the trigger is set to the clock and the data is probed. An equivalent representation would be to have an ideal data eye and show the distribution of the clocks in the center of the bit.

Defined this way, the eye diagram includes the voltage margins, the timing margins and the clock recovery function. By evaluating the probability of a bit occurring on the wrong side of the clock, the bit error rate of the link can be calculated. If an error does occur, the error checking and retry mechanism can be used to resend the packet. This provides a large statistical relief to the overall budget of the link.

1.7.2 Jitter Distributions

The types of distributions needed to adequately characterize the jitter distribution are the uniform, double delta and random Gaussian probability density functions. Double delta and Gaussian are well known distributions that are used often because they are very easy to convolve. Other distributions require numerical convolution. Uniform distributions can be used (Uj) to describe bounded random events that are equally probable but absolutely bounded.

Uniform jitter (Uj) has an equal likelihood of occurrence on a fixed interval. This is shown in Figure 19.

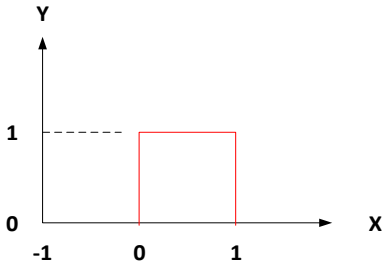

Figure 19: A Uniform Distribution Between 0 and 1

Convolving multiple Uj sources quickly approaches a bounded Gaussian distribution. This is shown in Figure 20 for different numbers of 10 ps uniform PDFs, convolved using frequency domain multiplication. For example, the result of 8 uniform PDFs, each of magnitude of 10 ps, gives a distribution that is bounded at +- 40 ps and is has the most likely occurrence at 0 ps.

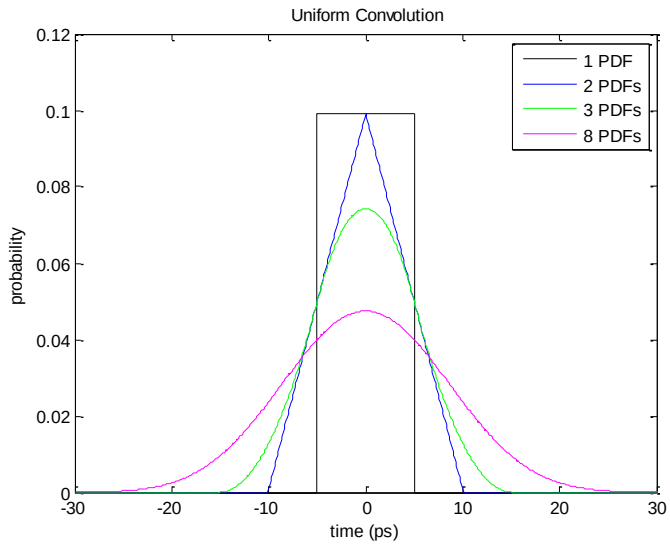

Figure 20: Multiple 10 ps Uj looks like bounded Rj

The complete distribution is found by assigning the proper jitter to the Uj, the Dj, and the Rj terms of all the jitter sources in the link gives a PDF that can be evaluated at the desired bit error rate.

The relationship between the voltage noise and delay is approximately linear for small variations. Regardless of the exact distribution, this distribution is bounded to the power delivery design targets. Each separate power node should be assigned its own bounded Uj distribution to represent this source of jitter.

As an example, a jitter budget for GDDR 5 is given in Table 1.

Table 1 : An example of a statistical budget for GDDR 5

	Jitter Item	Distribution	Rj (ps)	Dj (ps)	Uj
Rx					
	Vcc Jitter				1.00
	Rx Amplifier / Latch	Dj		15.00	
	Thermal	Rj	0.30		
	Offset induced jitter	Dj		8.33	
RxClock Tree					
	Vcc Jitter	Uj			2.00
	Clock Buffer Jitter Thermal	Rj	0.30		
	Clock Phase Mismatch and distribution mismatch	Dj		10.00	
PLL Rx Impact					
	PLL				
	PLL Core Jitter due to Vcc Noise	Uj			2.00
	PLL Thermal	Rj	2.00		
	PLL Digital feed through	Dj			8.00
Tx					
	Clock Buffer Vcc Noise	Uj			3.00
	Thermal Noise	Rj	0.30		
	Phase Mismatch	Dj		3.00	
	Predriver Error	Dj		4.00	
PLL Tx Impact					
	PLL				
	PLL Core Jitter due to Vcc Noise	Uj			2.00
	PLL Thermal	Rj	2.00		
	PLL Digital feed through	Dj			8.00
TxClock Tree					
	Vcc Jitter	Uj			2.00
	Clock Buffer Jitter Thermal	Rj	0.30		
	Clock Phase Mismatch and distribution mismatch	Dj		10.00	
Phase Interpolator (tx)					
	Vcc Jitter	Uj			5.00
	Thermal	Rj	0.30		
	Non-linearity	Dj		3.00	
	Step Size	Dj		5.00	
Channel					
	ISI	Uj			10
	Crosstalk	Uj			80
	MER budget	Uj			10
Total			2.90688837	58.33	133.00
EYE:					228.54

This budget can be classically convolved to get a total eye of 228 ps. Using numerical convolution with Uj distributions a more accurate prediction 205 ps is obtained, shown below in Figure 21.

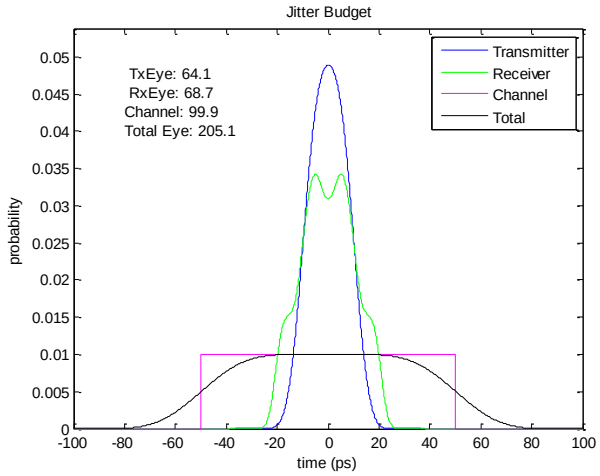

Figure 21: A numerically convolved jitter budget

In Figure 21, the different independent events given in table 1 are modeled in their respective probability density functions and the results are convolved to give the final distribution. In this case, deterministic jitter sources do not add linearly but instead convolve to the Gaussian distribution. The channel ISI data is kept as a uniform distribution, this could be mad more accurate by using the data pattern convolution method described later in the electrical noise section. Clearly, the more detailed the breakdown of independent sources, the more accurate the final PDF evaluation will be. Future work can address joint dependencies and static distributions due to manufacturing variations verses dynamic distributions due to dynamic fluctuations.

Convolution is achieved with numerical efficiently by using multiplication in the frequency domain. Due to the small amplitude of 1e-12 type of BER, the numerical round-off error is significant and must be carefully quantified.

2 Clock Recovery

All communication systems need a clock to sample the data. DDR sends a clock along with the data that the receiver uses to strobe multiple data bits. USB, SATA, and Ethernet expect the receiver to have knowledge of the ideal data rate and then look at the data itself to generate a clock that tracks the phase of the data. PCIe pseudo common clock mode sends a low speed clock that the receiver can use for tracking low frequency phase modulation in conjunction with data phase tracking. Clock recovery is a critical function for all communications. In this section a model is given for a digital clock recovery circuit that extracts the clock phase from the data.

2.1 Digital Clock Recovery

In today's high speed serial links the clock is extracted from the data by the Clock Recovery Circuit (CRC). This recovered clock is used to strobe the data and operates some of the logic in the receiver. The CRC is part of the larger Clock and Data Recovery circuit (CDR), sometimes referred to as the Data Recovery Circuit (DRC). Some functions of the CRC are to synthesize a clock in the absence of data transitions, multiply or divide the clock to different rates, generate multiple clock phases, and track the incoming jitter in a well-controlled manner. The block diagram for the CDR is shown in Figure 22.

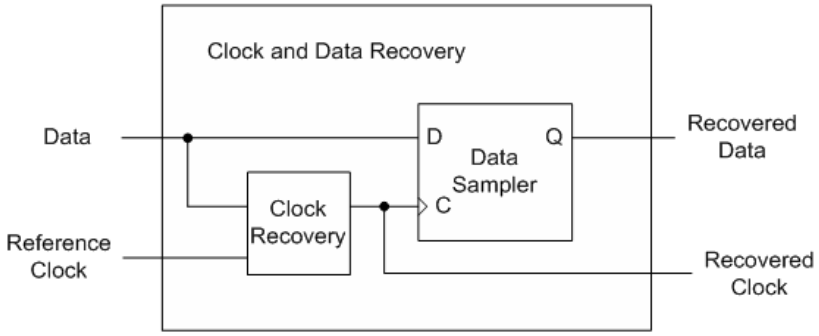

Figure 22: Clock and Data Recovery Circuit

A modified Phase Locked Loop (PLL) can be used to perform the clock recovery. The PLL has a proportional response to changes in the input phase of the data and tracks large amounts of jitter in a controlled manner. A modification to the PLL is to replace the noise sensitive VCO filter with a digital loop filter. The analysis of this loop is very similar to the analysis of the phase interpolator.

2.1.1 Phase Interpolating CDR

The first instance of the modern Phase Interpolator (PI) based CDR was published by Lee, Donnelly, et al of Rambus in 1994 [6] with an application to memory interfaces. The reasons cited as motivation for the new circuit was the noise immunity, lower voltage operation and fast lock time of the Phase Interpolator vs. the PLL. Simplistically, the PLL was replaced with two Delay Locked Loops (DLL) and a digitally controlled MUX. This enabled the DLL to track the input phase over arbitrarily large phase changes as long as the frequency difference was bounded and did not exceed the circuit's maximum slew rate.

In early 1997 the first usage of the term Phase Interpolation (PI) applied to a CDR was published by Sidiropoulos and Horowitz [7], also of Rambus. In 1998 Dally and

Poulton included a description of the PLL and PI CDR in their textbook on digital communications [8].

This section develops the model of the Alexander "bang-bang" style phase detector [21]. In particular, it sets out the challenges of tracking spread spectrum clocks (SSC) that have a 0.5% frequency difference that creates a relatively large low frequency phase jitter component. It covers the basic PI operation, the key performance requirements, and presents an implementation of the loop filter.

2.1.1.1 Phase Interpolation with Reference Clock

The phase interpolator block diagram is shown in Figure 23.

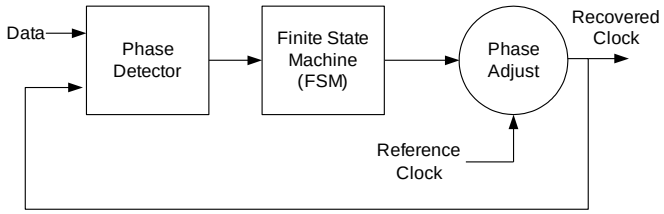

Figure 23: Phase Interpolator Block Diagram

The phase detector compares the phase of the recovered clock to the phase of the data. It generates early and late indicators to the Finite State Machine (FSM). The FSM is the digital logic that forms the control loop of the phase interpolator. The output of the FSM adjusts the phase of the reference clock to try to match the phase of the data. The frequency of the reference clock is bounded with respect to the data clock by the serial I/O specification, so the rate of the phase adjustment must be sufficient to overcome frequency differences between the data and the reference clock.

The maximum rate of change that can be generated by the FSM is a function of the step size and the rate of the indicators coming into the FSM, along with the gain of the FSM. The smallest possible step size is limited by the circuit tolerances of the particular process. Manufacturing tolerances in clock phase distribution, clock jitter due to noise and crosstalk, and other circuit errors typically are in the 1 ps range.

Large step sizes will track greater frequency deltas, since the phase will move farther with each adjustment. Since the phase detector is of the bang-bang type, the FSM will dither by at least one step. This dither is an error in the sampling location and is a consideration when choosing the maximum allowed step size. Smaller step sizes give smaller dither errors but cannot track larger frequency offsets.

The loop delay is the amount of time it takes from when the input to the phase adjuster changes to when the recovered clock actually moves. The loop delay introduces a pole into the phase interpolator loop that must be removed by under sampling or through filtering.

The rate of the indicators coming into the FSM is limited by the number of transitions in the data. Since this is a random quantity the FSM is not a regularly sampled system if it operates on every possible transition and cannot be easily modeled. For 8b10b encoding the worst case pattern has an edge at least once every 5 UI and an average edge density of 30% [9]. Other encoding, such as 64/66 generate similar edge bit densities but have no density guarantee.

2.1.1.2 Bang - Bang Type Phase Detector

The first step in recovering the clock from the data is to extract the phase from the input data stream relative to the reference clock. This is done by sampling the data at the edge and in the middle with the current recovered clock. The current recovered clock is the reference clock plus the phase adjustment from the phase interpolator. The result of this comparison is the data phase is detected to be either early or late relative to the sampling clock. This is accomplished by sampling the data in the middle of the bit and at the edge. This operation is shown in Figure 24 and summarized in Table 2.

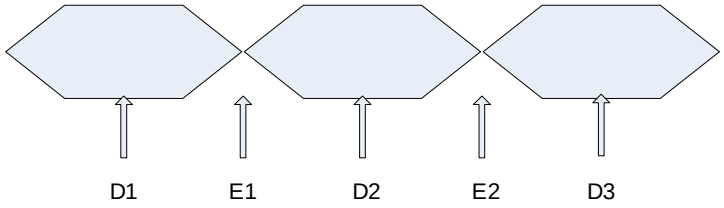

Figure 24: Sampling Edges and Data

Table 2: Early and Late Indicators

D1	E1	D2	Meaning
0	0	0	No Transition, do nothing
0	0	1	Sampling Early, move phase later
0	1	0	Phase is off by ½ bit time
0	1	1	Sampled Late, move the phase earlier
1	0	0	Sampled Late, move the phase earlier
1	0	1	Phase is off by ½ bit time
1	1	0	Sampling Early, move phase later
1	1	1	No Transition, do nothing

This phase detector can resolve a phase difference to within one half of a UI. If the sampling clocks move farther than one half of a UI the phase detector slips a bit. This

concept is exaggerated in Figure 25. In this figure, E1 is with D1 so the sample is early in B1. This continues for B2 and B3. In B4, E4 is the same as D4 and D5 so no information is given. But in B5 the edge sample E5 is with D6, so the sample is considered late. A bit slip occurs when D5 enters B4.

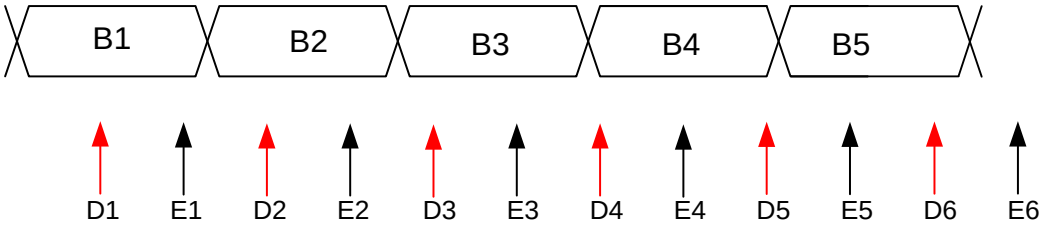

Figure 25: Bit Slip Illustration

With the requirement of SSC and assuming a 200 ps UI, the maximum difference is 1 ps per UI. With no phase adjustment a bit would be slipped once every 100 UI. If the PI is not capable of tracking this difference then the phase adjustment would oscillate at a period of 40 ns constantly slipping bits.

2.1.1.3 Note on Wrapped and Unwrapped Phase

The unwrapped phase is an accumulated phase that is greater than a period. The wrapped phase increases from 0 to UI and then resets to 0 and is always between 0 and the period.

Since the phase detector can only distinguish the phase difference to one half of a UI the phase detector operates on the wrapped phase. For most of our models we use the

unwrapped phase for ease of modeling. This does not change the model behavior as long as the total phase error between the sampling clock and the data never exceeds one half of a UI.

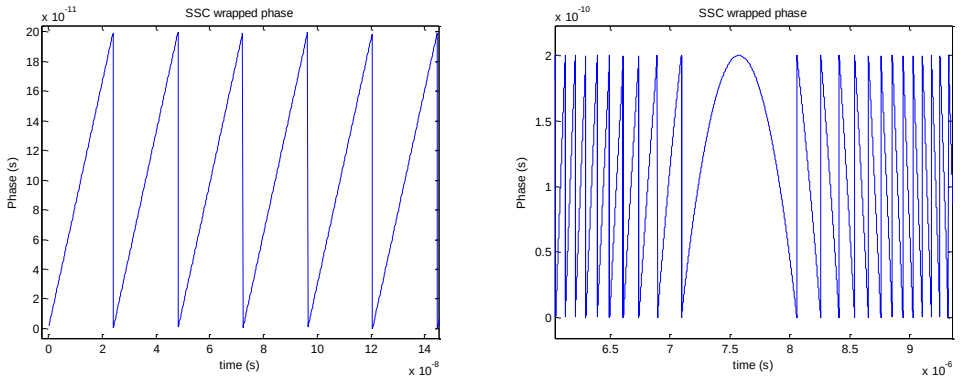

Figure 26: SSC Wrapped Phase near zero crossing (left) and near peak (right)

The wrapped phase for SSC is shown in Figure 26. Since the phase interpolator operates on the wrapped phase, it locks very quickly as long as it does not have an outer loop integral filter. The actual lock time depends on the step size, the under sample value and the current location of the SSC phase. At a maximum, the lock time will be less than 20 ns or the PI will not be able to track SSC.

Any digital filter in the FSM will use the wrapped phase, while the filter coefficients are valid only for the unwrapped phase. This means that the circuit must handle the numerical wrap to prevent the filter from overflowing.

2.1.1.4 Step Size of the Phase Adjuster

Practical limitations bound the maximum slew rate of the phase interpolator. Assuming that one step can be made every m UI, the maximum rate of change would be

$$\left(\frac{d}{dt}\phi\right)_{\max} = \frac{stepsize}{m*UI} \quad \text{2-1}$$

.

With a 5 ps step size, an m value of 5, a UI value of 200 ps then the maximum slew rate of the CDR is

$$\frac{5e-12}{5*200e-12} = 0.005\ s/s \quad \text{2-2}$$

and is not sufficient to track SSC. This is shown in Figure 27 using the unwrapped phase representation.

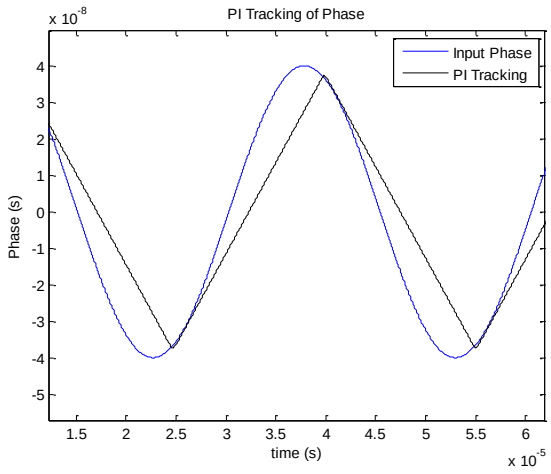

Figure 27: Tracking 40ns SSC with 5 ps step size, m=5

The maximum slew rate of a sinusoid is the sinusoid amplitude times its frequency (in radians). From this, the maximum amplitude is

$$A_{\max} = \frac{0.005}{2\pi f} \quad \text{2-3}$$

and is shown as in Figure 28. In this figure, the first horizontal line is an amplitude of 5 ps or one step, this occurs at about 100 MHz. The second horizontal line is 10 ps.

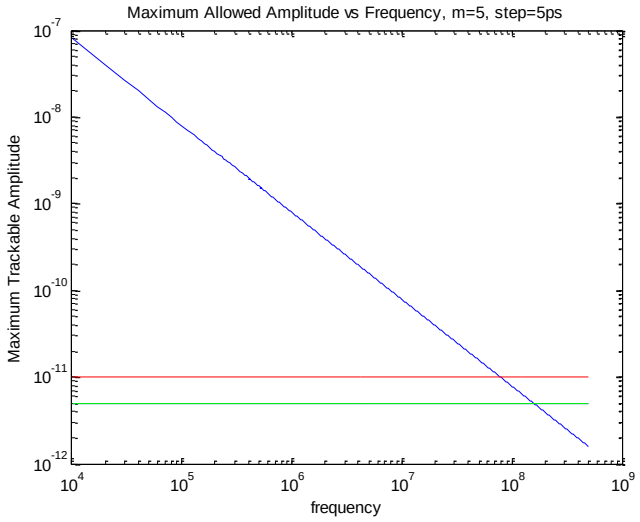

Figure 28: Maximum amplitude traced by a 5 ps step with m=5

With the SSC slew rate of $8.3\ \frac{m\sec}{\sec}$, a 200 ps UI, m=5 and using equation (1) gives the minimum step size of 8.3 ps. This is shown Figure 29.

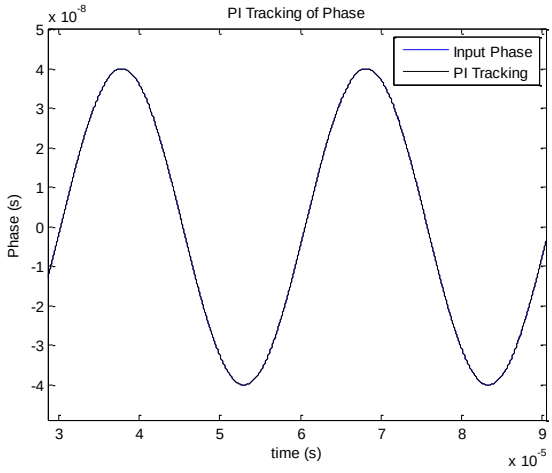

Figure 29: Minimum step size tracking of SSC

While at a step size of 8.2 ps the slew rate can no longer be completely tracked during the maximum. This small error cannot be seen on the tracking plot, but shows up in the error plot, shown in Figure 30.

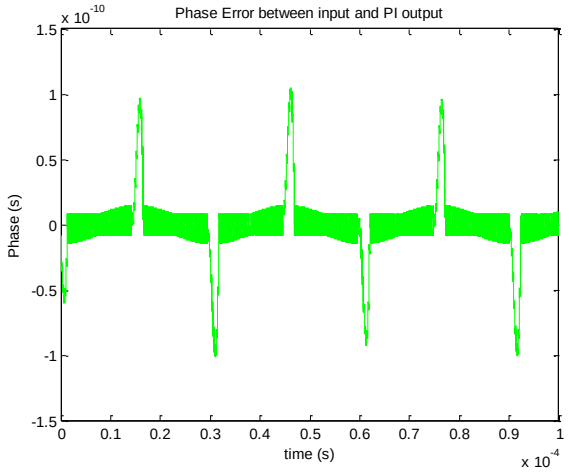

Figure 30: Error in steps between input SSC phase and output of PI

The larger the under sampling, the larger the step size required to be able to track SSC unless some form of gain is introduced into the control loop. For a 200 ps UI, the graph in Figure 31 provides the minimum step size required to track SSC as a function of the under sampling integer, m.

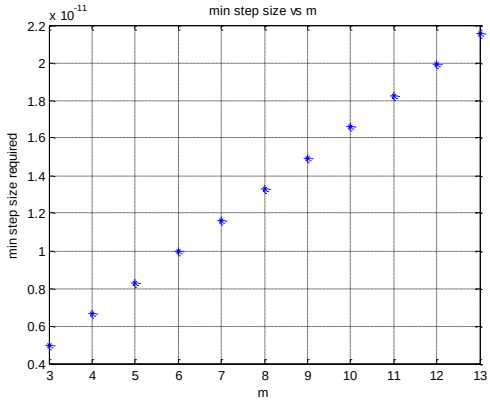

Figure 31: Minimum step size required to track SSC vs undersampling integer m

Although the larger step size does a better job of tracking SSC, it also causes a larger dither component that closes the eye. The amount of acceptable dithering at steady state sets the upper limit of the step size.

2.1.1.5 Early / Late indicators

Given the under sampling integer (m) and the nature of 8b10b, there are a variable number of indicators that arrive for each m UI. There are several ways to treat the extra indicators. One way is after the first indicator some circuits could be idled to save power. This is the simplest approach. A more complicated approach is to take an average of the indicators in order to get some type of magnitude. In the presence of sufficient random jitter the average of many samples will give an indication of magnitude as proposed in [10]. This is shown in Figure 32.

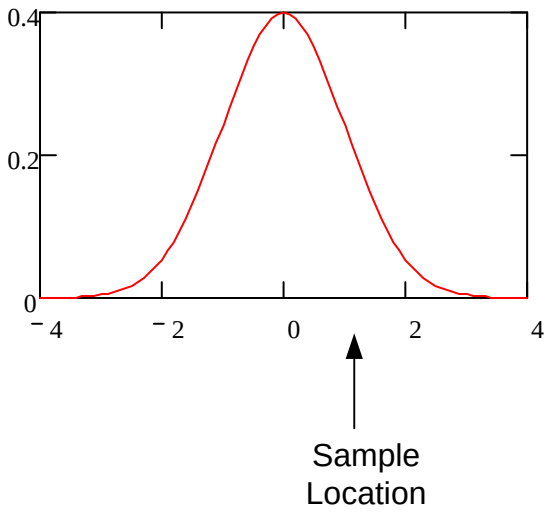

Figure 32: Sampling the tail of a random distribution

If the sample location is far to the right of the distribution, the value returned over m samples will be –m. If the sample location was to the far left of the distribution, the average returned over a large enough sample will be +m. If the sample location is at the center of the distribution, the value returned would be 0.

Sampling a Gaussian distribution at location sigma over m samples gives the result of m times the difference in the probability of left and right. The probability of left and right is given by the cumulative density function CDF(sigma),

2-4 m * (CDF(-k) – CDF(k))

where k is the location of the current sampling clock in terms of the sigma of the Rj distribution.

This averaging operation is

2-5 $$E_m = \frac{m}{k}\sum_{n=m}^{n+m} I_n$$

where E_m is the integer variable that represents the error between the clock and data, n is the bit rate, m is the decimation, k is the actual number of edges that occurred, and I_n is the indicator of early ,+1, and late, -1. As previously mentioned, E is only an integer variable in the region of sufficient random jitter. For 8b10b an edge is guaranteed once every 5 UI, and as often as every UI.

In order for the Rj to provide a magnitude indication the step size of the PI must be small relative to the sigma value of the Rj PDF. However, as the data rate for serial interfaces rises, the Rj component decreasing. For example, USB3.0 is considering a Rj component of only 3 ps. The total useable distribution would then be approximately 4*3 = +- 12 ps. With a 5 ps step size, the useable magnitude would be 2 or 3 at the most. Another risk to using averaging is when the Rj component is dominated by inter symbol interference (ISI). This means the PI can be fooled by a particular data pattern. The models that follow do not use averaging.

2.1.1.6 Spread Spectrum Tracking by a Digital Phase Interpolating CDR

The under-sampling integer, m, sets the Nyquist frequency of the tracking loop. Our model assumes that the output of the phase interpolator is updated every m^{th} UI. During the time of maximum rate of change of the SSC input, the difference in UI accumulates for the m cycles while the PI is not updating the output.

This accumulation is a source of tracking error. During the maximum frequency delta of the SSC the difference in UI is 0.005 UI. In the case of a 200 ps UI and an m

value of 10, the phase will change by 10 ps between updates. This will be positive for positive changes in slope, and negative for negative changes in slope giving an eye closure of 10 ps due simply to the lack of updates.

This jitter is called the breakthrough jitter. It may be compensated by various filter techniques that operate between updates; this is beyond the scope of this paper.

An example of breakthrough jitter with is shown in Figure 33.

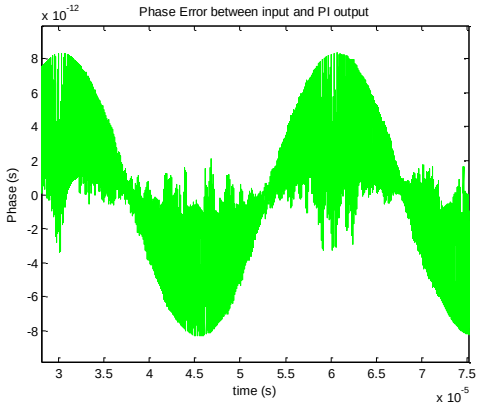

Figure 33: Breakthrough jitter with m=10, step = 5ps, 200 ps UI

2.1.1.7 Phase Adjustment

The phase adjustment is accomplished by back-to-back delay locked loops (DLL) as shown in Figure 34. A reference clock is delayed by small steps and a MUX allows selection of the proper phase. Two DLLs are used with one going forward in phase and one backwards in phase order to roll the phase smoothly through 360 degrees. A more detailed description of phase interpolator circuit can be found in the literature [11]. What is important for modeling purposes is that there is a number provided to the phase adjuster that ranges between 0 and N, where N is one UI divided by the step size.

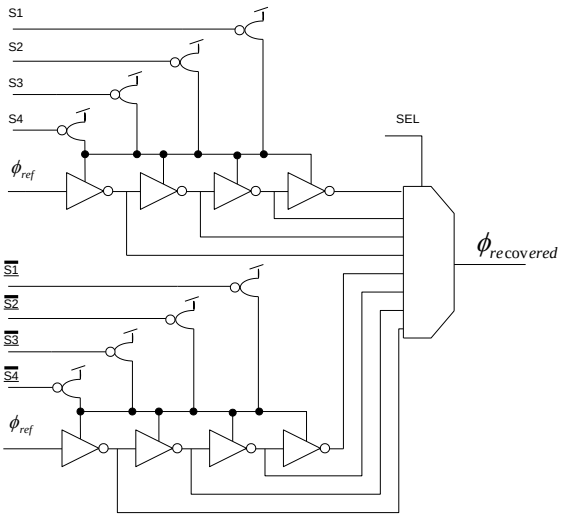

Figure 34: Schematic of a Phase Adjuster

2.1.1.8 Finite State Machine

The block diagram of the CDR is shown in Figure 35.

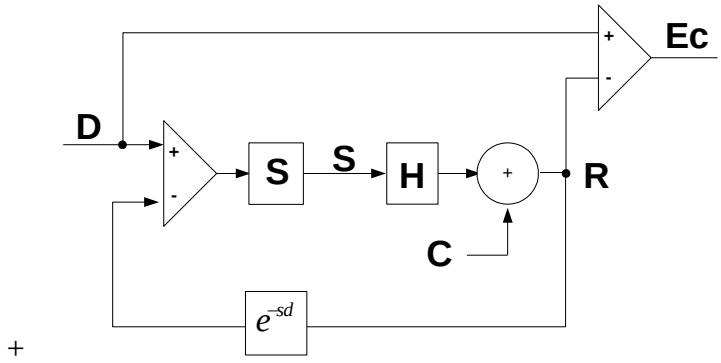

Figure 35: Block Diagram of the Clock and Data Recovery

The Nyquist frequency, *fn*, of the system is a function of the under sampling variable, m,

$$fn = \frac{1}{2mT} \quad \text{2-6}$$

Referring to Figure 35, the data phase D is compared to a delayed version of the recovered clock phase, R. Advance indicators increments the accumulator S and retard indicators decrements the accumulator S as appropriate. Filtering and gain is provided by the transfer function H, as appropriate. The output of H*S is a number between 0 and 2 pi and is added to the reference clock to generate the recovered clock, R. Since the phase jitter is always relative to the reference clock C, C can be chosen to be 0 and the double tracking amplitude.

The eye closure is shown as Ec and is the mismatch between the recovered clock and the data. This is the system transfer function that must be better than the clock recovery function given by the serial specification. Ideally Ec=0, meaning that the clock is sampling the data in the center of the bit. This control loop operates at the Nyquist frequency, given by equation 2-5, where m is the sub sampling integer.

2.1.1.9 Loop Delay

The loop delay in this circuit is **d** and is typically in the range of several UI. It is the amount of time it takes for the phase adjuster to change the setting of the current phase.

From the block diagram in Figure 35, the s domain response of D-C to R is:

$$R = \frac{D - C}{1 + H e^{-sd}} \qquad \text{2-7}$$

When D = C the eye closure is 0. This means that the recovered clock phase exactly matches the data phase and the eye opening is at a maximum. As previously discussed, we are interested in the response to the phase error between C and D. If the

loop filter, H, is 1 then the total response is completely determined by the delay in the loop. This is

$$H_{Ec} = \frac{Y}{D-C} = \frac{1}{1+e^{-sd}} \tag{2-8}$$

where H_{Ec} is the total transfer function. This can also be written

$$H_{Ec} = \frac{1}{e^{-\frac{sd}{2}}\left(e^{\frac{sd}{2}} + e^{\frac{-sd}{2}}\right)}, \tag{2-9}$$

and taking the frequency response, we find the magnitude as

$$\left|H_{Ec}\right| = \frac{1}{2\cos(\pi f d)} \tag{2-10}$$

where f ranges from 0 to Nyquist. With the Nyquist frequency normalized to 0.5, $\left|H_{Ec}\right|$ is plotted as a function of $f\,d$ in Figure 36.

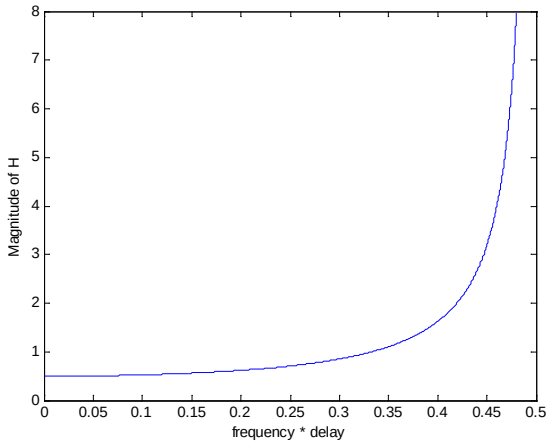

Figure 36: Magnitude vs Loop Delay * Nyquist

In this figure, f=0.5 represents the delay equal to the Nyquist period. This is the case when d = 2 m T. At f d = 0.25, the loop gain is less than one and the system is stable, this corresponds to d = m T. As f d approaches the pole at 0.5, the system will be unstable regardless of the input.

The minimum value of m required for stability is

$$m_{\min} = \frac{d}{T} \qquad \text{2-11}$$

For example, if the loop delay is 10 UI, m should be set to 10 or greater. This delay then limits the maximum frequency at which phase interpolator can operate. This is generally not a problem since the phase interpolator is not expected to track very high frequency jitter, and the loop delay should be below 5 UI. Figure 37 shows the tracking error vs the loop delay for two different values. The first plot shows stable operation, the second plot shows what happens when the pole from the delay is dominant and the tracking loop catastrophically fails. Errors are given in steps.

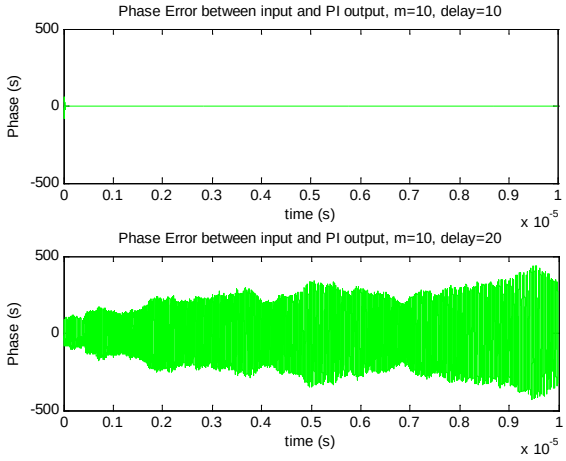

Figure 37: Stable and Unstable Loop Tracking

As discussed, larger values of m lower the ability of the loop to track SSC and the stability requirement conflicts with the SSC tracking requirement. This tradeoff can be optimized by adding loop gain to the tracking loop. Adding a proper loop filter to H can also attenuate the pole due to the loop delay.

2.1.1.10 Gain and Digital Filtering in the FSM loop

The PI block diagram is redrawn in Figure 38 showing the Z domain model. The reference clock C is no longer shown since we are free to pick the phase of C as 0, transferring any jitter on C to D. Just like a sigma delta converter, the output R is adjusted until it matches the input signal D by summing a sign indicator and not using a magnitude indicator. In a delta sigma ADC converter this adjustment happens quickly relative to the sample rate of the ADC itself, and once a stable value is found (a dither of 1 or less) then the output value is made available at the lower rate. For the PI, the output directly controls the phase select, so the PI is always subject to dither.

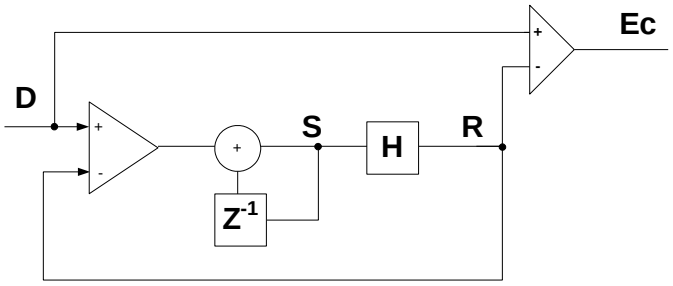

Figure 38: Z domain PI model

Some designs add an integral filter to compensate for the SSC phase jitter in an attempt to increase the low frequency tracking [12]. The alternative presented in Figure

38 is to use a simple digital filter, H. Then gain can be added and the response can be shaped as desired by appropriate choice of filter coefficients.

The total transfer function with the loop filter H in the z domain is

$$R = \frac{H\,D}{1 - z^{-1} + H\,z^{-d}} \qquad \text{2-12}$$

When H is implemented as a low pass IIR filter, the Z domain transfer function of H is

$$H = \frac{k\,a_0}{1 + b_1\,z^{-1}} \qquad \text{2-13}$$

.

k selects the gain of the loop, while the corner frequency of H is set by the choice of the coefficients a0 and b1.

With step=5ps, m =5, UI=200 ps, k = 1.8, fc=50e6, and the delay fd = 0.25, the transfer function of just H is shown in Figure 39.

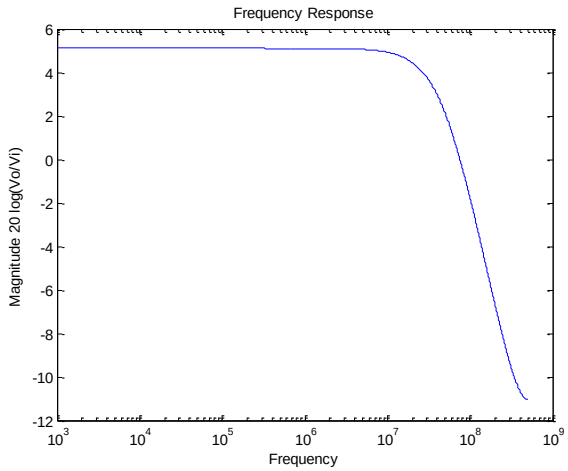

Figure 39: Transfer function of D to R

The eye closure, Ec, is given by

2-14 $$Ec = D - R$$

The transfer function of input jitter to eye closure with step=5ps, m = 5, fc = 50 Mhz, UI=200 ps, k = 1.8, and the delay fd = 0.25, is shown in Figure 39.

From Figure 39 it is clear that having H as a low pass counters the effects of the delay. This allows the circuit delay to increase and the overall system to remain stable provided H is a low pass.

Peaking occurs in the total transfer function as the corner frequency of H gets small. This is shown in Figure 40 for H=5 MHz. This type of behavior is not necessarily obvious and highlights the importance of solving the total system transfer function to avoid cases of peaking.

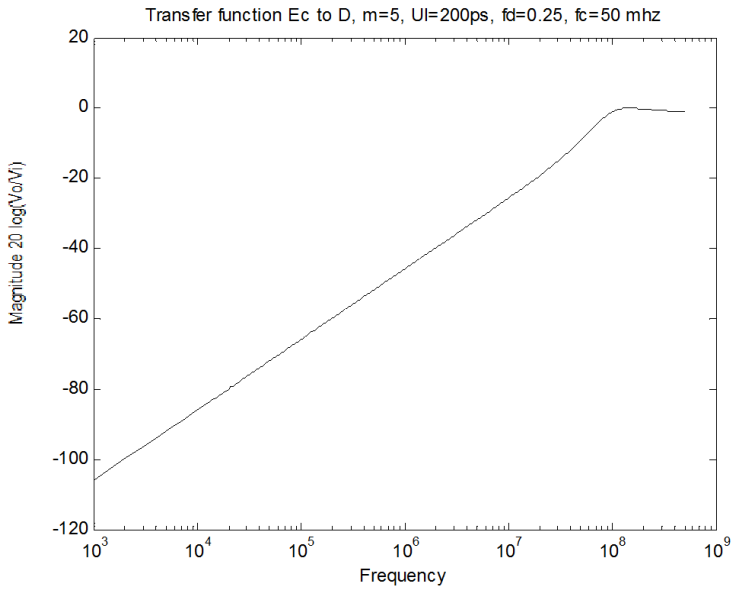

Figure 40: Transfer function of D to Eye Closure, EC

2.2 PLL Based CDR

The earliest types of CDR circuits were PLL based. The PLL CDR has the advantage of simple linear, time invariant system modeling but takes more power and area than the digital type of CDR. With modern digital processing techniques, the PLL and the phase interpolating type of circuits are blending. Digital filters are replacing analog circuits, offering greater noise immunity and programmable transfer functions. Regardless of the mechanism of the implementation, the major blocks of a PLL type of CDR remains the same.

A PLL CDR model is shown in Figure 41. The input data phase is compared to the output of a voltage controlled oscillator every time a data transition occurs. If the phase of the data is early, the PFD increases the frequency of the VCO to make the VCO phase better align with the data.

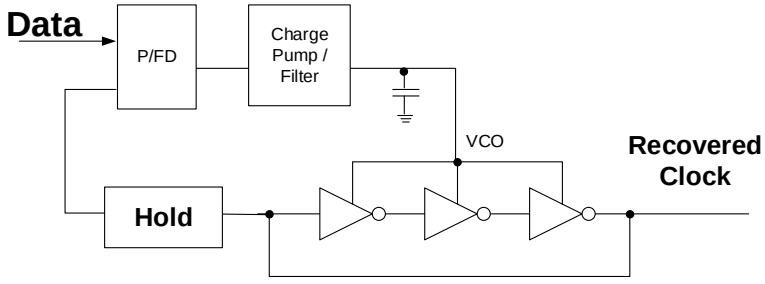

Figure 41: PLL Type CDR

The transfer function of a PLL type circuit is well known. When the PFD has missing data, the Nyquist frequency of the clock reduces to the lowest value defined by the longest run length of data. The worst case transfer function can then be modeled like a standard PLL with a second order response.

The second order model is based on the active proportional integration control loop with the transfer function given by:

2-15 $$H1(s) = \frac{2s\zeta\omega_n + \omega_n^2}{s^2 + 2s\zeta\omega_n + \omega_n^2}$$

In equation 2-14, ζ is the damping factor, and ω_n is the natural frequency. This function is not meant as a requirement for an implementation. It is used as a bounding function for modeling purposes to establish the lower limit for the -3 dB frequency and the maximum peaking. In practice, the actual transfer functions are likely third order and higher.

The translation between natural frequency ω_n and the -3 dB frequency is given by

2-16 $$\omega_{3dB} = \omega_n \sqrt{1 + 2\zeta^2 + \sqrt{(1 + 2\zeta^2)^2 + 1}}\,.$$

2.3 CDR Noise

The CDR loop is a source of noise due to either the digital jitter or due to the low frequency tracking response. For example, with m=5, step = 5ps, UI = 200ps, fc=50Mhz, and k=1.8, the tracking performance of 40 ns of phase jitter is +- 10 ps as shown in Figure 42.

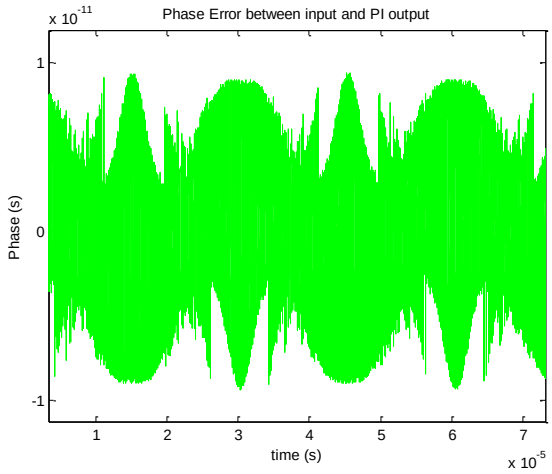

Figure 42: Tracking error with 80 ns of SSC

The entire tracking loop is given in the following Matlab program code:

```
for n=start_n:N
   if mod(n,decimate) == 0                    % update every mth UI
      S=S+sign(d(n)-r(n-delay));
      r(n)= (b1*r(n-decimate) + k*a0*S);    % apply the IIR filter
   else
      r(n)=r(n-1);   % otherwise just hold the value
   end
end
```

Examples of the tracking performance are shown in Figure 43 to Figure 45.

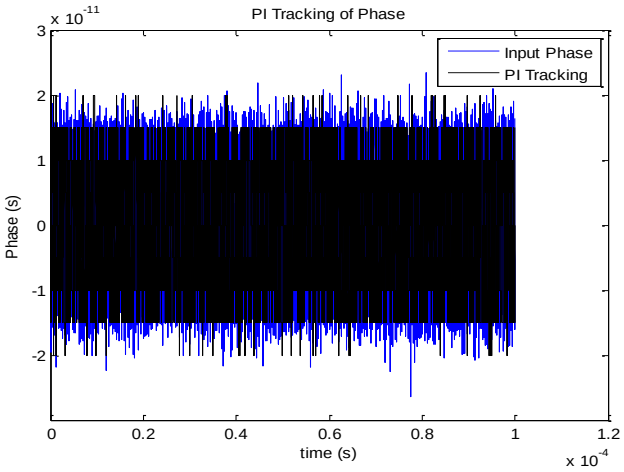

Figure 43: Tracking random jitter

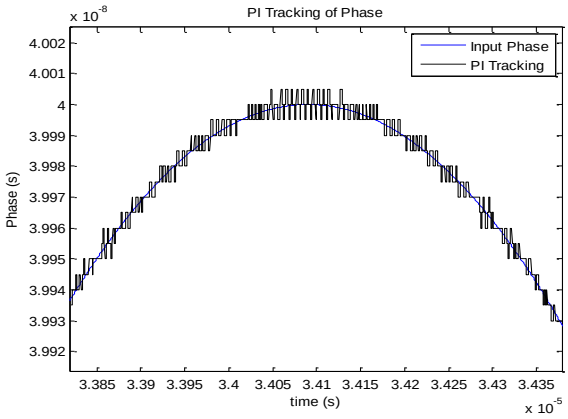

Figure 44: Tracking SSC at the top

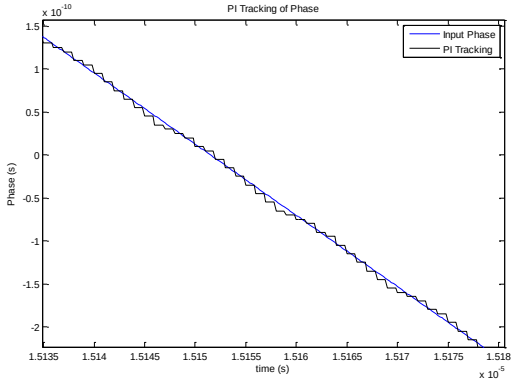

Figure 45: Tracking SSC through the maximum slew

2.4 Serial Specifications and the CDR

High speed serial specifications, such as PCI Express, USB3.0 and SATA, indirectly specify the minimum performance required by the CDR. They do this by specifying the filter function that is used to measure the jitter from the transmitter and receiver. The filter function is a high pass filter, rejecting any low frequency components and passing high frequency jitter. This is shown in Figure 46.

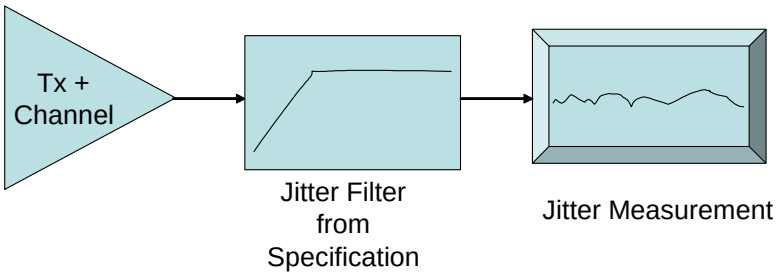

Figure 46: Measurement Schematic

It is implied that the CDR must operate as well as the filter function at tracking jitter. This means the CDR total jitter transfer function of the circuit must be less than the jitter filter at all points in order to meet the specification requirement. Since the specification is given as a frequency domain requirement, it is essential that the circuit be modeled in the frequency domain to ensure it meets the specification.

2.5 CDR Summary

The details of the phase tracking loop have been given, along with performance parameters that are critical to tracking SSC with a PI filter. Slew rate limitations have been given and the limits of step size relative to the sampling rate. The PI has been made linear by modeling it like a Sigma Delta ADC. Undersampling has been used to ensure a

linear response, and transfer functions have been given for one instance of a first order digital filter. The importance of delay has been demonstrated and the delay transfer function has been given. The parameters for avoiding the pole caused by the delay have been given.

This model highlights the slew rate as a limit of a digital tracking loop with a fixed step size. In order to bound the behavior of the phase interpolator type of CDR a slew rate limit must be specified and designed.

One of the main sources of jitter in a CMOS clock distribution is the fluctuations of the power supply causing the delay through a gate to vary. This section models and quantifies this source, and provides a non-linear model that is needed for modeling sub ps effects. In this chapter we discuss different noise sources of jitter and provide models useful for the quantification of each source.

3.1.1 Delay through an Inverter

There are a number of sources of jitter in the clocking architecture. Many of these jitter sources, like clock duty cycle distortion, do not change during operation and are captured in the jitter budget. These are static jitter sources and are called deterministic jitter (dj). The main source of dynamic jitter is due to the voltage fluctuations on the transistors. The delay through the transistor increases as the voltage drops. This is shown in Figure 47 for two inverters:

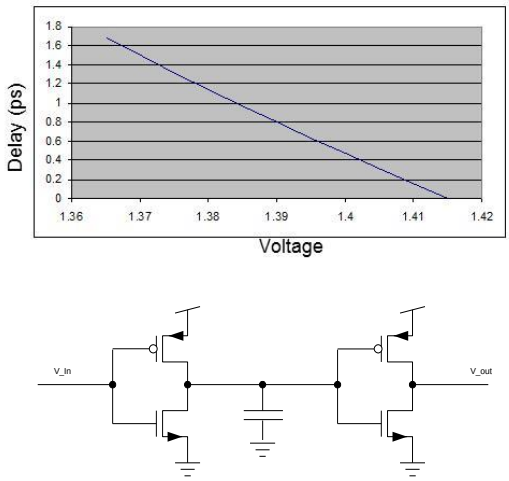

Figure 47: Inverter with Delay as Function of Voltage

The voltage variation seen by the inverters is due to current transients in the power delivery network (PDN). In order to calculate the jitter budget, the power delivery network must be specified.

3.1.2 Power Delivery Network (PDN) Model

The simplified PDN circuit is shown in Figure 48 :

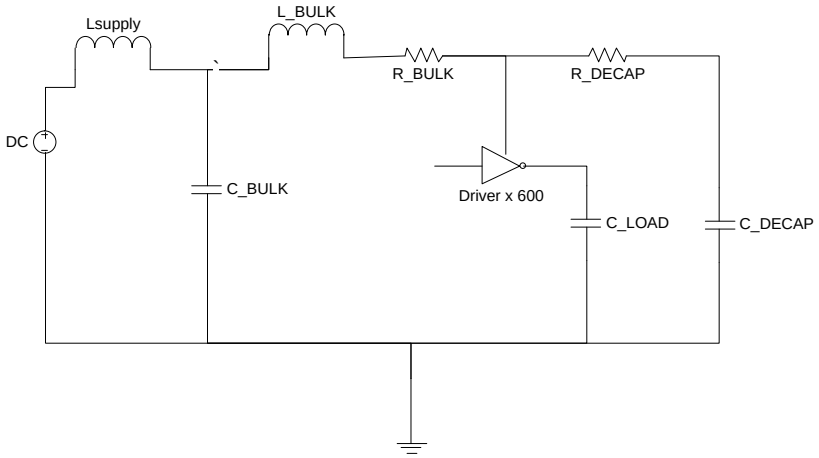

Figure 48: Power Delivery Diagram

The voltage regulator (DC) delivers power to the bulk decoupling capacitance that is placed as close to the chips as possible. The bulk capacitance is sufficiently large that it behaves as an ideal voltage source. From the bulk capacitance to the drivers, there is additional inductance and resistance shown as L_BULK and R_BULK above. On die decoupling capacitance is provided on the DRAM and the SOC, shown above as C_DECAP. Getting from this on die capacitors to the actual driver adds some resistance,

R_DECAP shown above. An example PDN response drawing current at 7 ma is shown in Figure 49:

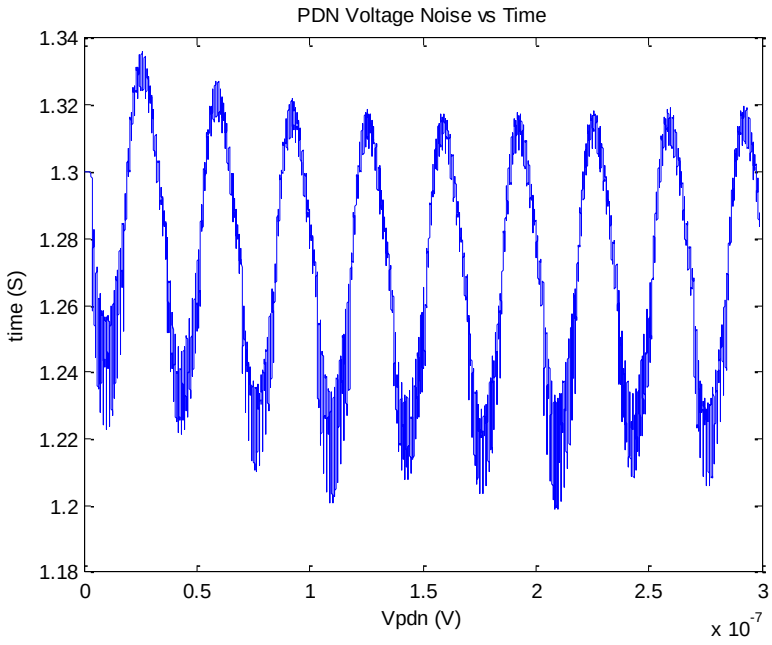

Figure 49: Power Delivery Noise

In the above picture, the current is drawn in a burst – idle – burst pattern that was chosen to maximize the resonance response. In addition to the resonant response, high frequency drops can be seen at the bit rate. These are the two different mechanisms of simultaneous switching noise (SSN) that contribute to voltage fluctuations on the PDN, I*R drop and charge sharing by the capacitor divider network of C_DECAP and C_LOAD.

First consider the I*R droop. The driver pulls current from the PDN when it drives the data lines from low to high. The inductance, L_BULK, prevents the bulk capacitance from instantaneously providing current; all of the current going to the driver must come from C_DECAP. This causes an I*R drop across R_DECAP. This voltage

drop is a function of the current being drawn times the resistance, R_DECAP. Higher strength drivers cause a larger voltage drop since they draw more current. The duration of the IR drop is the amount of time it takes to charge C_LOAD.

The current spike duration is approximately the same as the rising edge of the DQ bit. This current spike is short compared to the resonance of the LRC network (L_BULK, (R_DECAP+R_BULK), and C_DECAP) and acts as an impulse to the resonance circuit and starts the LRC network oscillating. This IR drop is seen in Figure 50, a zoom in of the above response.

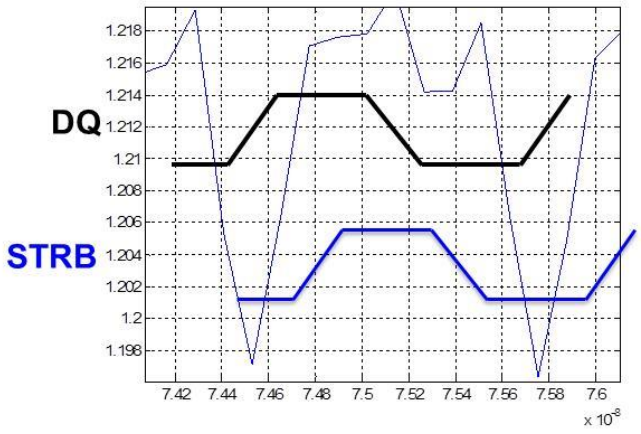

Figure 50: PDN Voltage Drop Due to Switching Circuits

In Figure 50, the PDN voltage is shown along with the clock and data timing. This is a case of the maximum bits switching at the same time and drawing the maximum amount of current. The switching of the DQ causes the PDN to droop, and then recover before the edge of the strobe. This voltage droop will slow down the edge of the DQ and push it to the center, shrinking the data bit.

However, with only 1 bit switching there is effectively no droop and there is no timing shift of the DQ bits. This difference in timing pushout between all bits switching and one bit switching is the contribution of the PDN to the eye closure.

For example, approximately 600 signals can switch at any one time. With the maximum current of 12 ma per driver, this gives a total current spike of 7.2 amps. A 10 mohm resistance will give a 72 mv drop across the resistor.

The second mechanism is a charge sharing effect that drives the resonance response. From the initial conditions of 0, there is no current through the inductor L_BULK. During the initial switching all the charge that goes into the loads, C_LOAD, must be supplied by the capacitor C_BULK. In this case, after C_LOAD has charged, regardless of the driver strength, the voltage on the drivers will be

3-1 $$V_{final} = V_0 \left(\frac{C_{BULK}}{C_{LOAD}+C_{BULK}}\right)$$

With a total bulk decoupling capacitance of 20 nF and the worst case load capacitance of 2.4 pf per pin, 600 pins, the charge sharing can leave the bulk decoupling capacitor discharged by 80 mv. The ringing that occurs proceeds in steps of 80 mv per each UI, until the inductor current starts to recharge the capacitor. Both the voltage drop due to IR and the charge sharing must be considered when establishing the minimum decoupling capacitance needed. The combination of charge sharing and IR drop is seen in Figure 51.

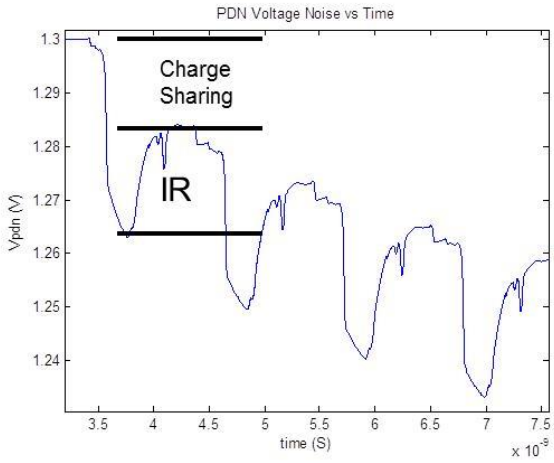

Figure 51: Power Delivery Voltage Drops

The resonance response is seen to be a pulse response of the charge sharing network. The first bit causes a voltage drop across the inductor, L_BULK, and current starts to flow. The current flowing in L_BULK, after the first bit is given by

3-2 $$I = \frac{1}{L}\int (V_o - V_f)dt,$$

where $V_o - V_f$ is the voltage across the inductor.

However, since the charge sharing is always common to both the DQ and the strobe, it does not cause a relative shift in the timing and does not contribute to eye closure. For the HBM clocking architecture, with less than 1 UI delay path difference in the DQ and strobe, the main contribution to eye closure is the IR drop from the decoupling capacitors to the driver buffers.

3.1.3 Clock Distribution

Clocks must be distributed across the component. Due to the capacitance of the distribution line, the clock must be buffered across the die to maintain an appropriate edge rate. This causes power supply noise to convert to phase jitter. In this section we

present the model for non-linear conversion and integrate it with the PCIe Gen 4 reference clock jitter transfer model. This allows accurate prediction of the eye margin degradation for the PCIe serial bus as a function of the Vdd noise profile. In turn, this allows an understanding of the Vdd budget required given the timing allocation. As an example in the following, we use a 20 mv PDN noise source at 3.5 MHz and show that it generates less than 1 ps of jitter to the PCIe G4 serial link. In the following, a clock distribution block is made up of N inverters. An example in Figure 52 is for N=4, the entire block is the sum of the 4 inverters,

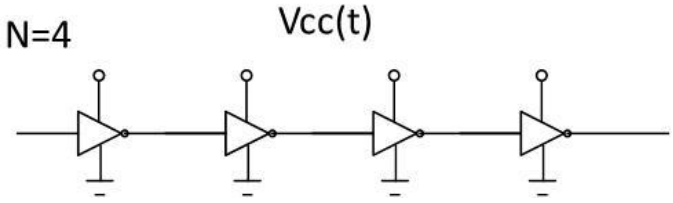

Figure 52: Clock Distribution Block

3.1.4 Power Noise Variance to Delay Variance

A transistor's switching time is a function of the transistor voltage. The alpha model is used to approximate this delay of the entire block as a linear relationship, where alpha is defined as in [13]:

$$\alpha = \frac{dt/DELAY}{dV/VDDQ} \quad \text{3-3}$$

Using this definition of α, the jitter through a block is given by [14]

$$\Delta t_d(t) = \alpha \int_{t'=t}^{t+D} \frac{\Delta V_{dd}(t')}{V_{dd}(nom)} dt' \quad \text{3-4}$$

where D is the *nominal* delay through the block, $\Delta V_{dd}(t')$ is the AC voltage variation, and $V_{dd}(nom)$ is the nominal voltage on the block. This integral gives the average voltage on the inverter over the delay time.

3.1.5 Non-Linear PDN-Jitter Model

In [14], alpha is given as a composite function of voltage and time, $\alpha(\Delta V_{dd}(t))$ and cannot be brought outside of the integral. A more accurate approach is to use a composite function of delay and time, $D(V(t))$, and integrate over the total delay, not using alpha.

The following example uses a block of 10 inverters on a 14 nm silicon process. The delay as a function of V at the output of the block is fit with a second order polynomial using the least squares method, giving the following relationship between the voltage and the block delay shown in Figure 53.

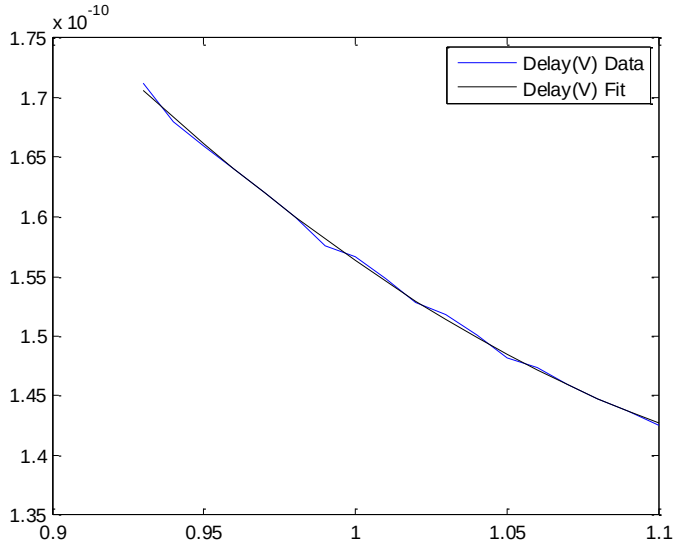

Figure 53: Nonlinear Delay through 10 Inverters

The polynomial giving the relationship between delay and voltage is:

3-5 $$D(V) = AV^2 + BV + C$$

For the example plotted above on a 14nm silicon process, with 10 inverters gives

3-6 A= 398.4 ps / V^2

3-7 B= - 973.1 ps / V

3-8 C= 731.1 ps

Equation 4-5 gives the jitter as a function of time. For example, with Vdd = 0.94V, a 20 mv sinusoid at 3.5 MHz, (2) gives the delay as a function of time as shown in Figure 54.

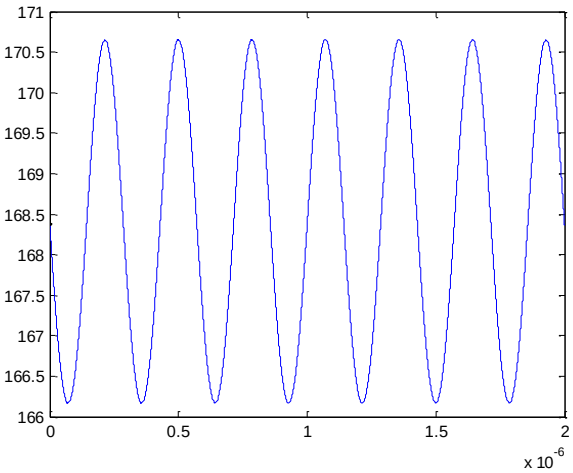

Figure 54: Delay Due to a 20 mv AC Source

The peak to peak jitter from this 20 mv, 3.5 MHz voltage noise is 4.59 ps, closely matching the TIE peak to peak jitter from simulation of 4.51 ps. Using the linear model of (1) gives 5.2 ps.

The integral in equation 4-4 accounts for the average voltage on the block, assuming a continuous variation of the voltage to the jitter is occurring inside the block. Our approach is to break the block into N inverters, and to average the delay on each inverter over the inverter switch time. This approach captures the non-linearity of the voltage to jitter conversion and quantizes the effects of inverters in the chain. We do this by summing the jitter from each gate at the appropriate times,

N*InstantaneousDelay(t) = D1(t) + D2 (t-nd) + D3 (t-2nd)+ …

Integrating gives the total delay, written as

$$D'(t) = \sum_{n=1}^{N} \frac{1}{N} \int_{\tau=t}^{\tau=t+\Delta} D(\tau - n\Delta)\, d\tau \qquad \text{3-9}$$

where N is the number of inverters and Δ is the delay through a single inverter. The high frequency filtering effect of this approach is similar to both [12] and [13], and is plotted in Figure 55 as the peak – peak jitter, where the integration effect is seen in the roll-off of the jitter at frequencies above 200 MHz.

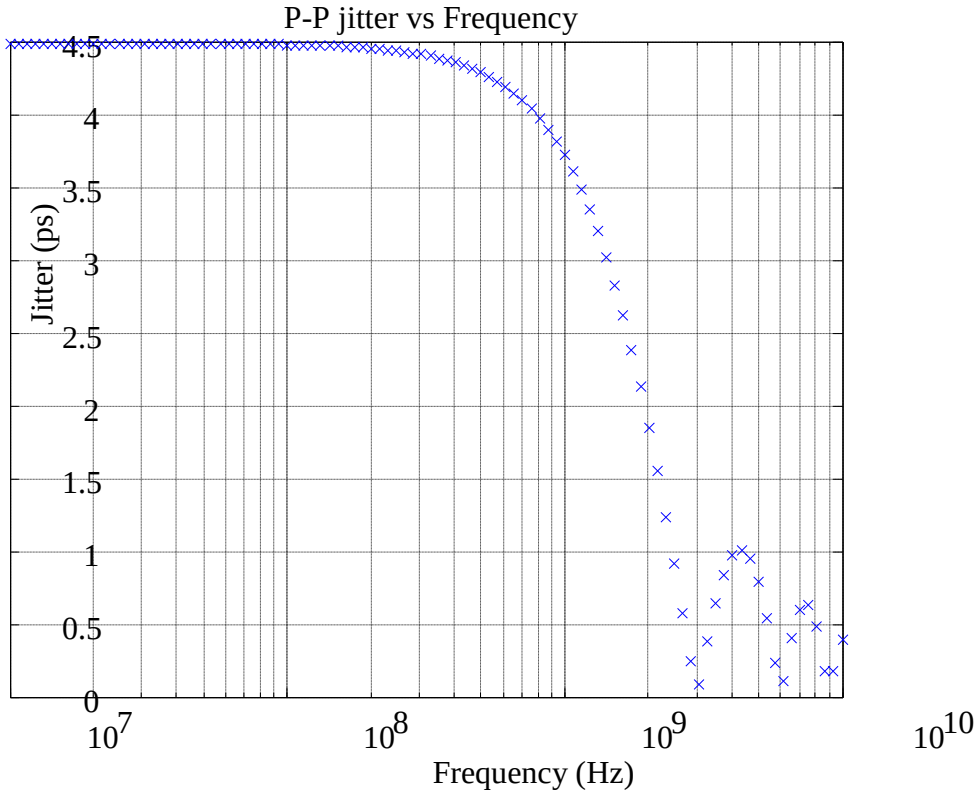

Figure 55: Jitter vs Noise Frequency

The next filter to be applied to the jitter is the clock sampling period itself. The clock period sets the Nyquist jitter frequency, and the jitter output is sampled to include the aliasing effects for the reference clock being modeled. In other words, the voltage and the delay functions are continuous, but the edge position of the clock is a sampled

variable. The frequency response of sampling aliases the clock frequency SSO noise to DC and can be discarded for most communication links as this is normally trained out. This means that the SSO noise on the PDN induced by the buffers themselves can be ignored as long as the minimum Vdd of the PDN is not violated.

For the 3.5 MHz PDN noise, undersampling and integration does not attenuate the jitter. For higher frequency noise or higher frequency clocks the under sampling and integration effects can be significant.

3.2 Summary of Voltage to Jitter Conversion

The analysis method presented allows us to quickly vary the PDN voltage, the PDN noise, the PLL bandwidths, and the CDR transfer functions to get the impact of the EBG PDN jitter on the PCIe G4 budget. In this approach it is easy to use a capture of the actual PDN noise and get the impact on any interface. It is likely the PDN jitter for the 8 mm of clock distribution can be absorbed in the reference clock budget. The next steps are to get the correct H3 transfer function and to engage the PDN budgets to ensure the impact of the PDN jitter can be absorbed in the reference clock budget.

4 Electrical Noise to Jitter

In this section we provide models for electrical noise processes that cause time delay variation. The largest such source is platform crosstalk coupling onto the clock. Crosstalk is the capacitive and magnetic flux coupling between an aggressor trace and a victim trace (the clock). The signal on the aggressor trace will cause a voltage to occur on the victim trace. This voltage conversion method is demonstrated with crosstalk, but the same process would also account for radiated coupling, thermal noise, or any other type of electrical noise.

4.1 Crosstalk Conversion to Jitter

Crosstalk impact on jitter is an important topic to closing the platform clocking budget as it is a significant contributor to eye closure at the receiver. It is also a complicated jitter model that involves both the electrical propagation and phase propagation. While this section uses crosstalk as the electrical noise contribution, other noise sources, like oscilloscope noise or thermal amplifier noise, could be treated in a similar manner.

The process to calculate the crosstalk impact on clock jitter is summarized the following:

1. Create a clock with an edge rate that is expected to be the minimum edge rate for that particular clock. Alternative is to capture an actual source clock output on a quiet platform (all aggressors are inactive).

2. Measure or extract from simulation the step response of the electrical noise of nearby aggressors on the platform
3. Create a data pattern and sum the crosstalk step response for every change of bit, with a different pattern for each aggressor to create a total electrical noise signal.
4. Electrically filter the crosstalk noise by the appropriate receiver filter function, emulating the receiver bandwidth.
5. Add the electrical noise to the clock and convert the resulting signal to periods.
6. Process the periods accordingly to ensure the budget, with the crosstalk noise, is met at the sampling latch.

4.1.1 Electrical Noise to Jitter

When noise is added to a clock, it can change the time at which the clock crosses the clock switching threshold. This causes the sampling position to move, thus it creates jitter. The amount of time changed depends both on the magnitude and duration of the electrical noise, and also depends on the slew rate of the clock. The noise to time relationship is

$$\text{TimeShift}(t) = N(t) / (dv/dt) \tag{4-1}$$

where dv/dt is the edge rate of the clock, N(t) is the electrical voltage being added as noise and Timeshift(t) is the difference between where the clock threshold ideally should have been to where it actually occurred, i.e., the phase jitter.

As the noise can have both polarities, the maximum time variation will be 2* Noise / (dv/dt). This can be used to check the crosstalk results based on the amplitude of the step response. For example, if the step response is 20 mv and the slew rate is 1v/ns,

the maximum period jitter is 40 ps since the two edges of a period can be shifted in opposite directions. The maximum phase jitter (due to crosstalk) is 20 ps, as the phase is bounded from one period to another and can never accumulate for more than 1 period.

4.1.2 Aggressor Step Response

The aggressor step response can be measured in simulation using HSPICE or another suitable simulator. In some instances it may be possible to drive the aggressor with a step and measure the actual response. An example of a step response from a PCIe G4 aggressor is shown in Figure 56. All voltages are differential voltages.

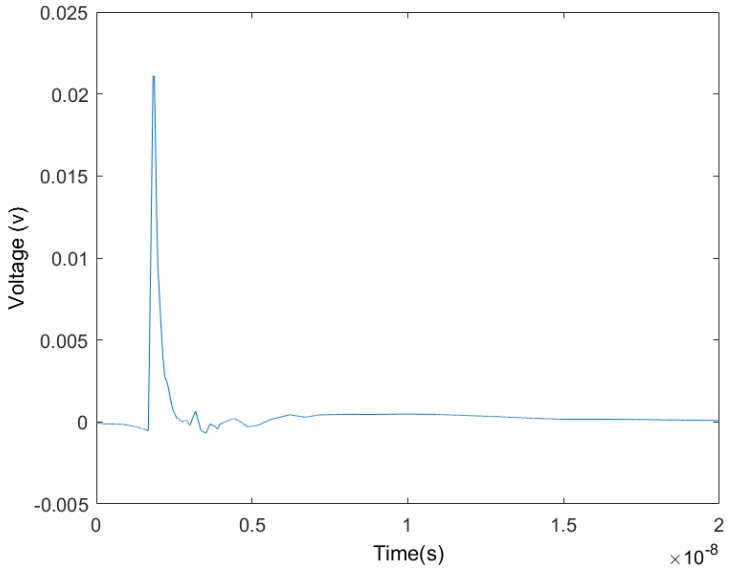

Figure 56: Crosstalk Differential Step Response

4.1.3 Crosstalk Voltage Record

The next step is to create a data stream at the aggressor bit rate. The bit stream is used to determine where exactly positive crosstalk pulses go, and where negative pulses go. This is shown in Figure 57.

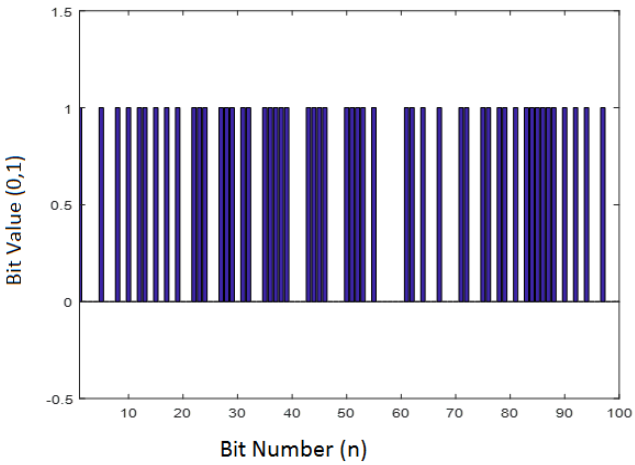

Figure 57: Data pattern from PCIe G4 LFSR

All of the transitions are then summed. The crosstalk step on rising edges is added and the crosstalk step on falling edges is subtracted. This creates the crosstalk electrical noise, shown in Figure 58, that would be present on the bus as measured with an oscilloscope.

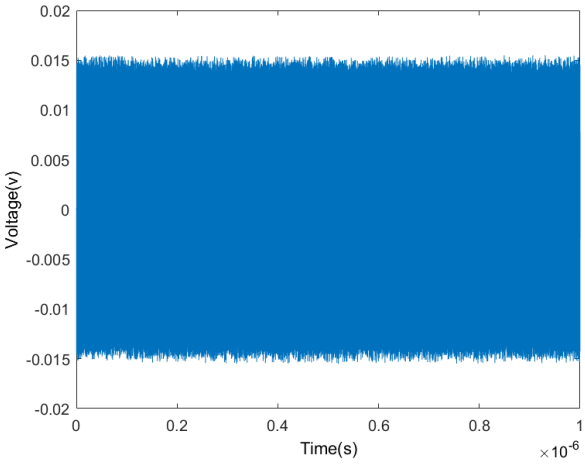

Figure 58: Crosstalk electrical signal

4.1.4 Crosstalk to Periods

The crosstalk is added to a clock signal and filtered through an appropriate receiver model. Below the filter is a 1 GHz first order op amp filter, with unity gain. The clock signal can be synthesized, or it can be a measured clock. Synthetic clocks have the advantage of controllable edge rate and perfect phase. This is useful for demonstrating trends. An example of crosstalk added to a synthetic clock is shown in Figure 59.

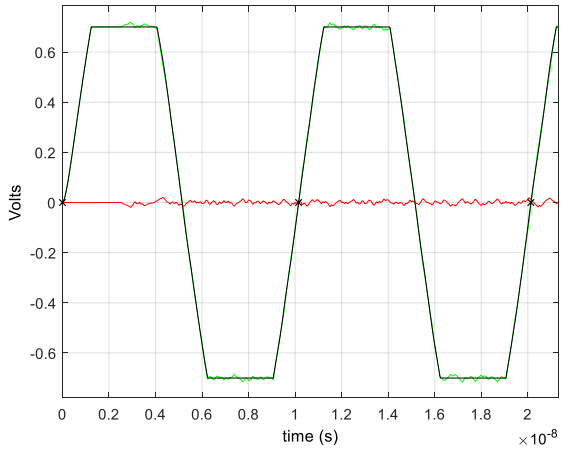

Figure 59: Synthetic clock with crosstalk, 1 v/ns slew

4.1.5 Crosstalk Conversion to Periods

The signal is next analyzed for the zero crossings, shown as ‘x’ in Figure 59. The zero crossings are calculated by linear interpolation between the sample points to the zero crossing voltage. This gives the period jitter. Integrating gives the total phase jitter. This is shown in Figure 60.

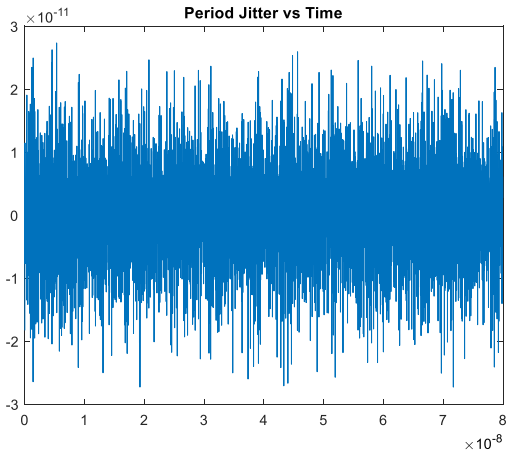

Figure 60: Period Jitter from xtalk, perfect clock

The period jitter is integrated to get the phase jitter, shown in Figure 61. As expected for a synthetic clock, the phase jitter is bounded to less than the period jitter.

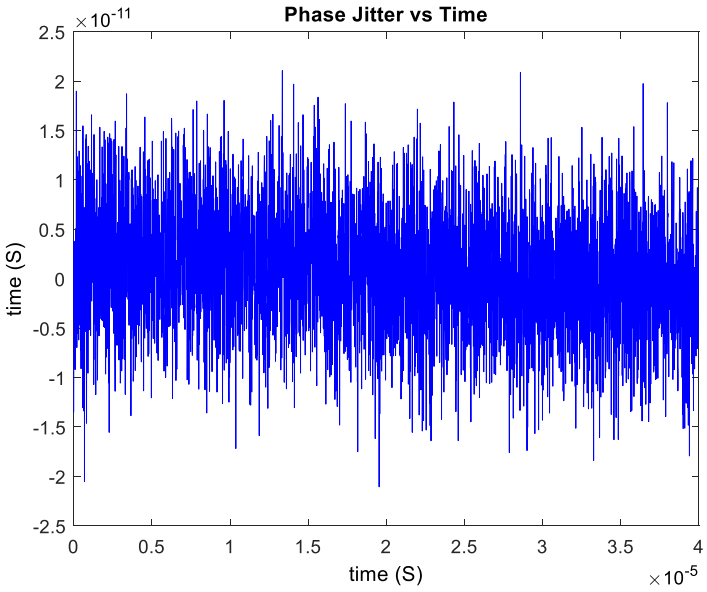

Figure 61: Phase Jitter due to xtalk

The phase jitter is then processed through the appropriate system transfer function. Since crosstalk is single sided jitter, the PCIe transfer function is not the

appropriate transfer function to measure the impact of the crosstalk induced jitter at the receiver. The correct transfer function is single sided, or

4-2 $Y = N * H1*H3$

The PCIe transfer function for the induced crosstalk is shown in Figure 62.

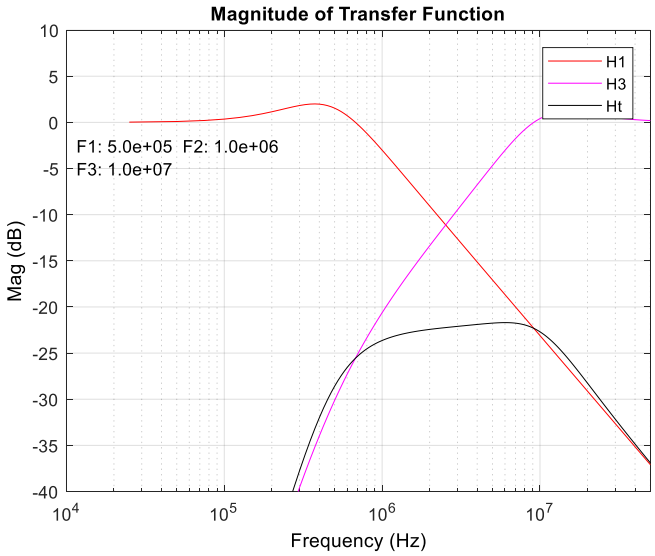

Figure 62: Transfer function of one of the PCIe G4 single sided

In the frequency domain, the input jitter spectrum is shown along with the eye closure spectrum in Figure 63.

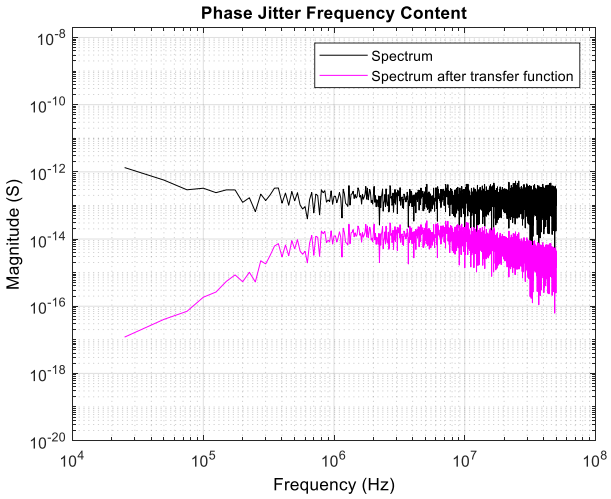

Figure 63: Phase Jitter Frequency Content

Taking the inverse transform of the spectrum after the transfer function has been applied gives the eye closure at the receiver. This is shown in Figure 64.

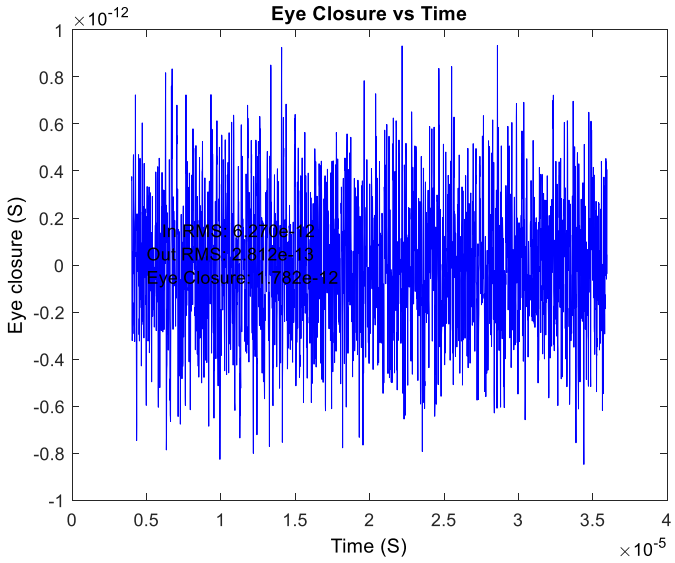

Figure 64: Eye Closure at Receiver, PCIe G4, 281fs RMS closure

4.2 Summary

Adding electrical noise to a signal causes the signal to shift in time. The time shift at the sampling threshold causes jitter in the signal, and the magnitude of the time shift is dependent on the slope of the signal at the threshold time. The conversion from the electrical voltage to the time shift is bounded for crosstalk electrical noise, as the crosstalk noise is bounded due to the AC coupled nature crosstalk noise.

By convolving the crosstalk step response with the appropriate data pattern, the proper distribution is coupled into the timing jitter. Due to the system transfer function, described in detail in the next section, this timing jitter is attenuated due to PLL filtering and CDR tracking. The final full system impact of the electrical noise can be determined by first calculating the timing jitter as described in this section, then applying the system models, described in section 5.

5 System Models for TIE Analysis of Clocking Jitter

This section provides the entire system transfer function models of the jitter for different types of communication architectures. These models quantify how noise sources previously discussed propagate to the sampling latch and cause timing error at the sampling latch. The result of this work is the time domain calculation of the time interval error between the sampling clock and the ideal location of the data for the different types of communication links.

These models calculate the final error at the sampling latch due to the phase jitter propagation through the system. There are three key components for these models. First, there are the previously discussed phase jitter sources, such as PDN induced jitter, electrical noise induced jitter, and tracking error from the CDR block. Second, there is a phase jitter transfer function through the phase lock loops used to multiply the reference clock frequency that operates on the phase jitter in the system. This transfer function is modeled to an acceptable degree of accuracy as a second-order transfer function. These phase lock loops also generate their own random phase jitter, well-modeled as a Gaussian random jitter variable. Finally, there is a transport and circuit delay that must be accounted for as the phase jitter propagates through the entire system. In the following sections, models are provided for the jitter to eye closure transfer function for different communication architectures.

5.1 Parallel TIE

The complete clocking architecture for a source synchronous interface, like DDR memory, is shown in Figure 65. As seen in this figure, a reference clock with initial phase jitter X is connected to a PLL that increases the frequency to the data rate. The read and write strobes are phase aligned by DLLs in the controller. This alignment is done after reset using a loopback training algorithm. The strobes originate as close to the data as possible, so as much jitter as possible is common to both the strobe and the data. The timing paths differ from the data paths by the delay inserted by the DLL. The eye closure due to the source jitter is given by the difference transfer function, where the difference is the DLL insertion delay.

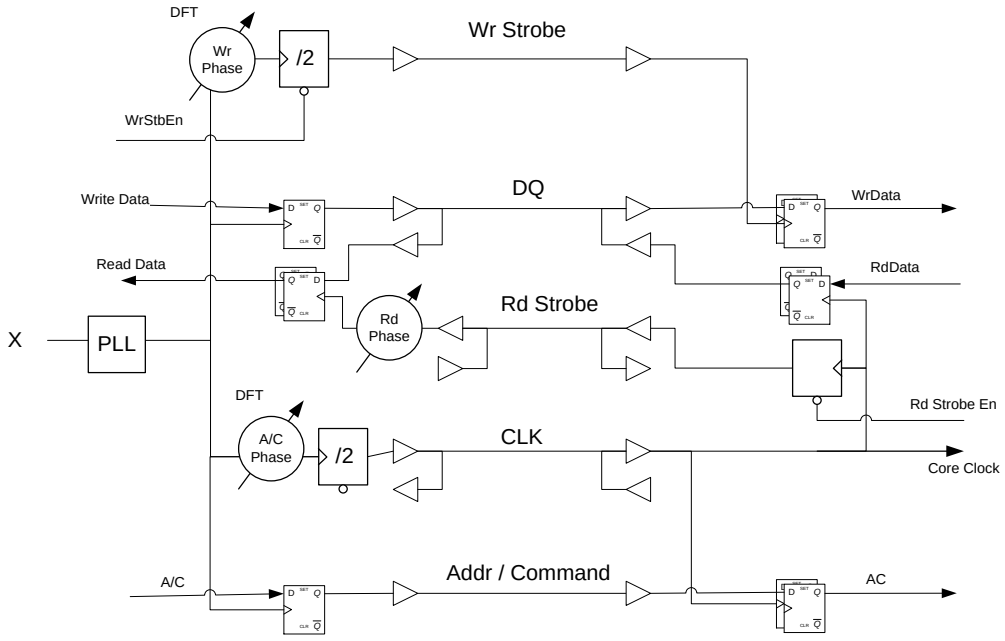

Figure 65: DDR Clocking Architecture

This are actually three separate sections to this architecture: the data write from the memory controller to the DRAM, the data read from the DRAM, and the writing of

address and command (A/C) to the DRAM. The A/C path is a copy of the data write path, but normally at ½ the rate as the data write path. The same analysis applies to both.

5.1.1 A/C and DDR Write Path

The write path can be looked at separately to simplify the analysis. The simplified write schematic is shown in Figure 66.

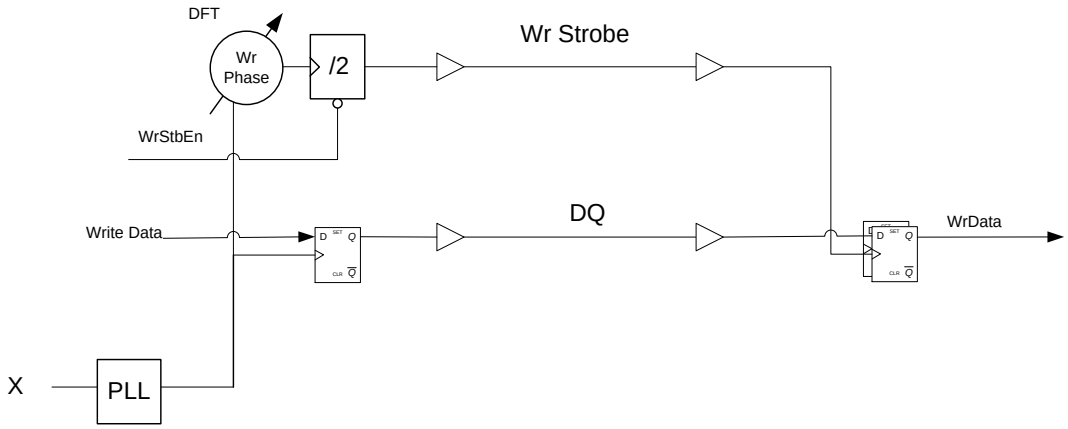

Figure 66: Simplified Forwarded Clock Path Jitter Model

In this simple circuit, the clock generated by the phase lock loop (PLL) drives both the write strobe and the data bits. There is an optional Wr Phase adjust circuit that could be designed as a delay locked loop (DLL) or some other adjustable delay circuit. This circuit delays the clock by 90 degrees to align the rising and falling edges of the write strobe to the center of the DQ. The phase transfer of the circuit can be modeled as shown in Figure 67.

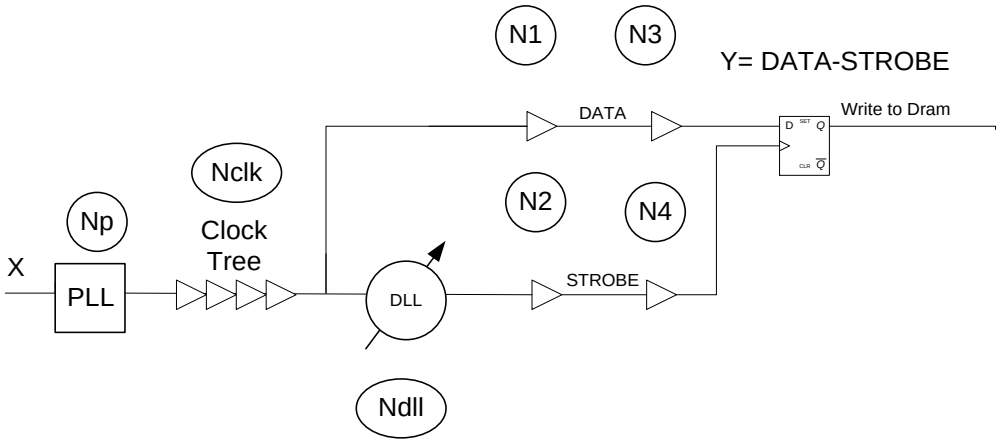

Figure 67: Clock Forwarded Math Model with Jitter Sources

In this figure, the phase of a source clock X is filtered by a PLL. Phase noise from the PLL (Np) is added to X. The output of the PLL goes to a clock tree, which adds additional phase noise Nclk to the signal. At this point the clock splits and goes to both the strobe and the data paths. The eye closure, Y, is the phase of the data minus the phase of the strobe. However, the strobe path has been delayed by some amount.

This means that

5-1 $$Y(s) = \left(X(s) + Noise(s)\right)\left(e^{-s\,dclock(t)} - e^{-s\,ddata(t)}\right)$$

where dclock(t) is the time dependent (due to PDN noise) delay path on the clock side and ddata(t) is the time dependent delay path on the data side. Equation 5-1 is read as "the eye closure due to jitter is the phase of the data minus the phase difference of the clock times the input noise in that band" [3]. If the clock and data were perfectly phase matched, the noise would be correlated and would not contribute to the effective jitter. However, if the clock and data path are not matched, the clock tree, PLL, DLL, and all

the associated noise will cause eye closure at the receiver by a difference function, described previously for N cycle jitter as a comb filter.

5.1.2 DDR Read Path

The read path uses the system clock as the reference to asynchronously launch both the DQ and the strobe from the DRAM to the memory controller (MC). The memory controller aligns the read strobe to the data and latches the DQ. The phase transfer through the read path is shown in Figure 68.

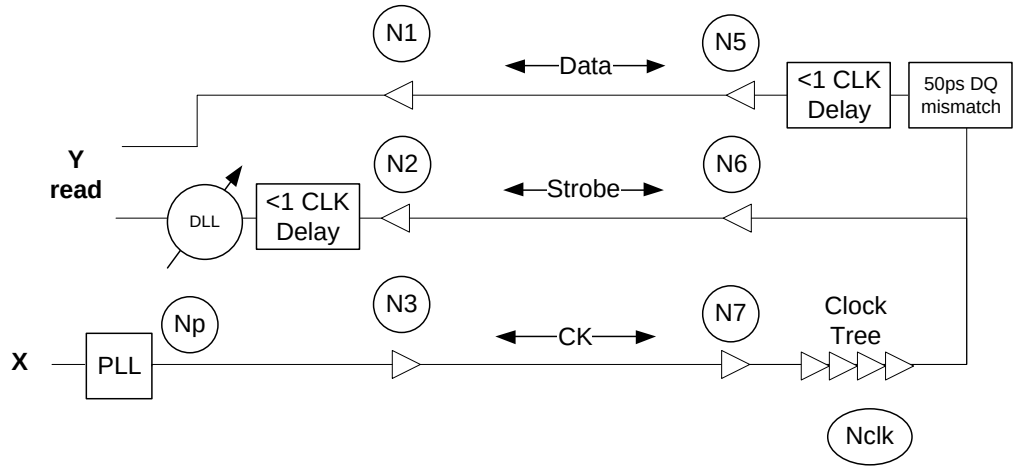

Figure 68: Clock Forwarded Read Math Model with Noise Sources

This path has the same effective jitter and jitter transfer function as the write path. The source jitter, X, will be larger than the write path due to the distribution of the clock through the channel and the extra buffering from the channel driver.

5.2 Mesochronous Serial Link TIE

The clocking architecture chosen by PCI Express is mesochronous. The PCI Express clock jitter specification is meant to allocate a portion of the Unit Interval (UI)

budget to the effects of the reference clock jitter assuming the CDR uses the phase information in the reference clock for part of the CDR operation. The clocking block diagram for the mesochronous architecture used by PCI Express is shown in Figure 69.

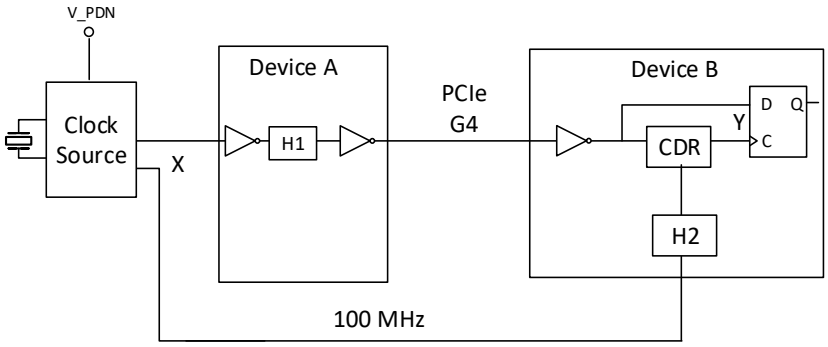

Figure 69: Chip to Chip Mesochronous Clocking used by PCI Express

To simplify the analysis, this is redrawn as functional phase blocks, shown in Figure 70.

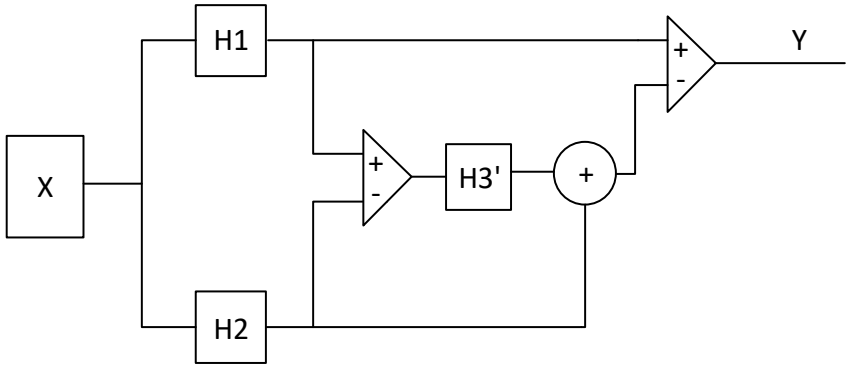

Figure 70: PCI Express Clock Jitter Transfer Model

From this figure, the difference at the sampling latch, Y, is calculated as

5-2 $$Y = X\{H_1 - [(H_1 - H_2)\,H_3{}' + H_2]\}$$

where X is the source jitter, H1 and H2 are the multiplying PLL transfer functions, and H3' is a digital CDR previously described. The CDR provides the phase difference to the

low pass filter, H3', and that filtered difference is added to the phase of the receiver clock to generate the sampling clock. Note that H1, H2, and H3' are all low pass filters. Using the substitution H3=1-H3', the high pass equivalent transfer function simplifies to

5-3 $$Y = X(H_1 - H_2)\, H_3.$$

This can be drawn as shown in Figure 71 and is the common representation used in PCIe, where the high pass function H3 is the l-H3' of the actual CDR response.

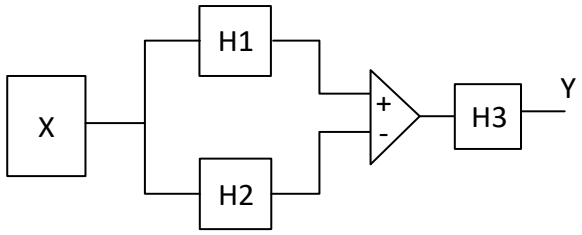

Figure 71: Simplified Mesochronous Transfer Function

Using this model, the eye closure due to source jitter can be calculated numerically from a measured reference clock, X. Worst-case PLL transfer functions and transmission line propagation is used to calculate the eye closure. The CDR model of the receiver is assumed to be a phase interpolating CDR with a tracking response of 10 MHz. This budget is just for the reference clock effects. Other items, like PLL thermal jitter, on die PDN jitter, and many other jitter sources are budgeted in the transmitter, the receiver, or the channel jitter budget.

Figure *72* shows a more complicated clocking architecture with the insertion of clock distribution buffers and crosstalk noise. These buffers are used to increase the number of clocks available and optionally can contain filtering PLLs with their own transfer function and inline noise sources. The PLL transfer function equation

is available from the buffer manufacturer's data sheet. The PLLs will add their own random phase noise to the incoming clock jitter. A typical buffer with a VCO based PLL architecture will add a phase noise that is reasonably modeled by a 15 ps RMS Gaussian jitter source. An LC based PLL architecture in HBW mode is modeled by a 2 ps RMS Gaussian jitter source. In pass through mode, the buffers add a small amount of jitter due to the thermal noise of the buffer and any power supply noise that causes delay variation through the buffer. This is on the order of 0.2 ps RMS.

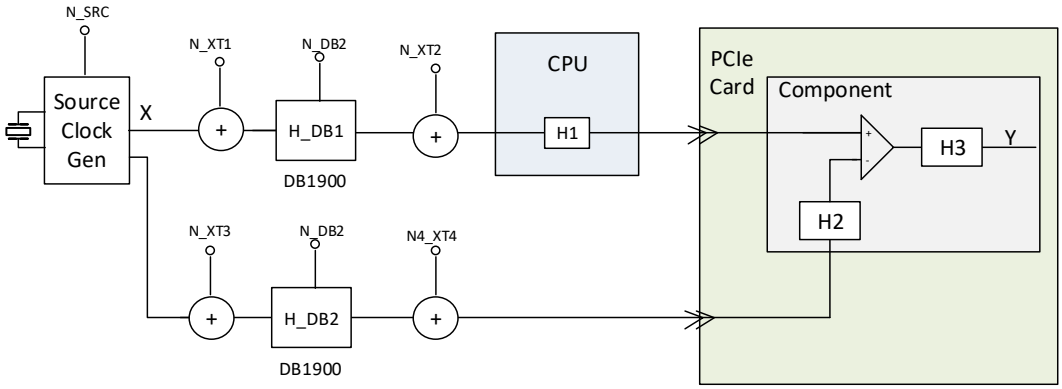

Figure 72: Example PCIe Clocking Diagram with Distribution Buffering

The complete eye closure for this model, moving from left to right and top to bottom, is modeled by

5-4 $$Y = \left\{\left((X + N_{XT1})H_{DB1} + N_{DB_1} + N_{XT2}\right)H_1 - \left((X + N_{XT3})H_{DB2} + N_{DB2} + N_{XT1}\right)H_2\right\}H_3$$

with proper the addition of the transport delays, as discussed in a following section. If the clock distribution buffers do not employ PLLs, $H_{DBx} = 1$,meaning all frequencies are passed through.

This can be written as the sum of the common mode jitter that is due to the reference clock, X, and the single sided jitter due to the uncorrelated noise sources, where

5-5 $$Y_{common} = X(H_{DB1}H_1 - H_{DB2}H_2)H_3$$

and

5-6 $$Y_{single} = [(N_{XT1}H_{DB1} + N_{DB1} + N_{XT2})\,H_1 + (N_{XT3}H_{DB2} + N_{DB1} + N_{XT2})H_2]\,H_3$$

The noise added by the clock distribution buffers and the crosstalk is single-sided noise in that it is only on one side or the other. The sum of the noise is filtered by the multiplying PLLs, H_1 and H_2, and tracked by the CDR, H_3.

This means the effective contribution to eye closure from a single-sided noise is measured by a single-sided transfer function. For example, the phase jitter contribution in bypass mode is measured after being filtered by the following transfer function:

5-7 $$H_t = H_1 * H_{CDR}$$

where

5-8 $$H_1 = \frac{2s\zeta\omega_n + \omega_n^2}{s^2 + 2s\zeta\omega_n + \omega_n^2}$$

and

5-9 $$H_{CDR} = \frac{s}{s + \omega_c}$$

with

$$\zeta = 0.54$$

$$\omega_n = \frac{2\pi 5e6}{\sqrt{1+2\zeta^2+\sqrt{(1+2\zeta^2)^2+1}}}$$

5-10(a-c) $$\omega_c = 2\pi 10e6$$

This function represents the highest bandwidth PLL expected along with the lowest bandwidth CDR tracking function for PCIe standards.

This transfer function is shown in Figure 73:

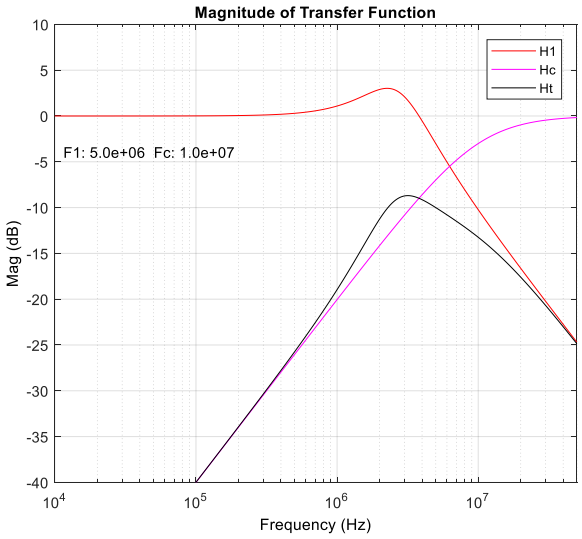

Figure 73: Single Sided Jitter Transfer

Jitter noise between 1 MHz and 30 MHz is transferred as eye closure, with the maximum sensitivity in the 2-4 mhz range transferred at -8dB. When there are two random sources in each path, the total jitter will be the root sum square (RSS) of the two filtered independent sources. Per the PCIe Gen4 specification the total jitter allowed at Y is 0.7 ps RMS.

5.2.1 Transport Delay Effects

Increasing the delay difference serves to de-correlate the common phase jitter components, X, depending on the frequency of the jitter component. Higher frequencies experience greater de-correlation than lower frequencies for a given delay difference. The delay difference arises from the different electrical lengths of the clock distribution and the data link, plus any inherent PLL or buffer delays. The worst-case delay difference must be used when evaluating the final eye closure. The default delay difference, per the PCIe specification, is 12 ns.

5.2.2 Mesochronous Model Example

To restate the assumptions, these results are not the complete numerical budget. The missing ingredients are the correct delays, the crosstalk, and the worst-case clock sources. In addition, the clock sources used (PCH typical clock) include a significant amount of scope noise that needs to be accounted for and is discussed in a later section. We use ~100k periods for this analysis that was capture at 20 ps / samples and linearly interpolated for the zero crossings.

Additionally, the PCIe transfer function defines many cases and combinations of PLL bandwidths and peaking values that must be checked in order to establish clocking compliance. This study uses a combination of min / max BW and peaking that is often, though not always, the worst case. Unless otherwise stated, the delay difference is 12 ns.

The baseline case is the PCIe standard of Y=X(H1-H2)H3. We use a typical SSC clock capture that has been converted to periods.

The baseline phase jitter that comes from the clock generator, X, with SSC is shown in Figure 74:

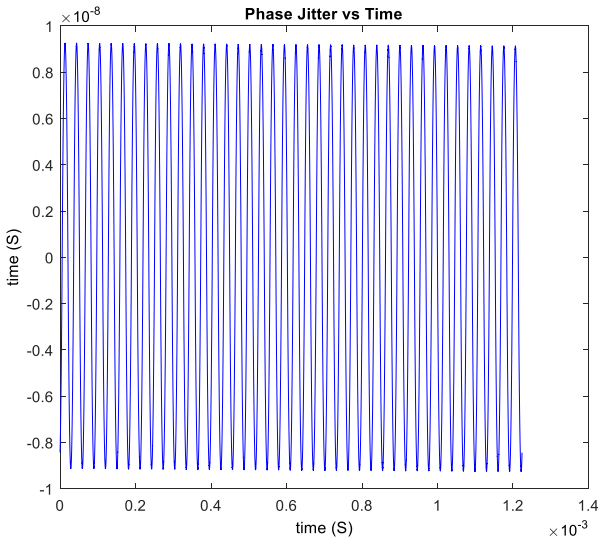

Figure 74: Unfiltered Phase Jitter with SSC

The transfer functions of X to Y is shown in Figure 75.

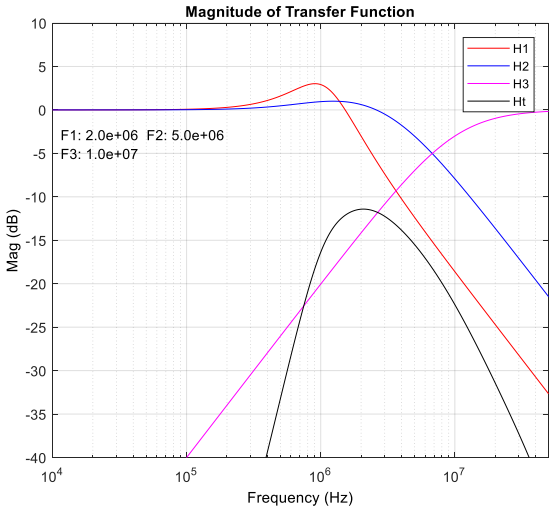

Figure 75: Transfer Function of Input Jitter to Eye Closure

Where the total transfer function is shown as the bandpass filter. This shows the frequency response of the transfer function is most sensitive in the 500 KHz to 30 MHz. Lower frequencies are tracked both by the difference function and by the CDR. Higher frequency jitter is rejected by the PLLs.

The results can be seen in the frequency domain. The input jitter is attenuated by the transfer function, SSC is clearly visible at 33 KHz with harmonics extending to 1 MHz. This is shown in Figure 76.

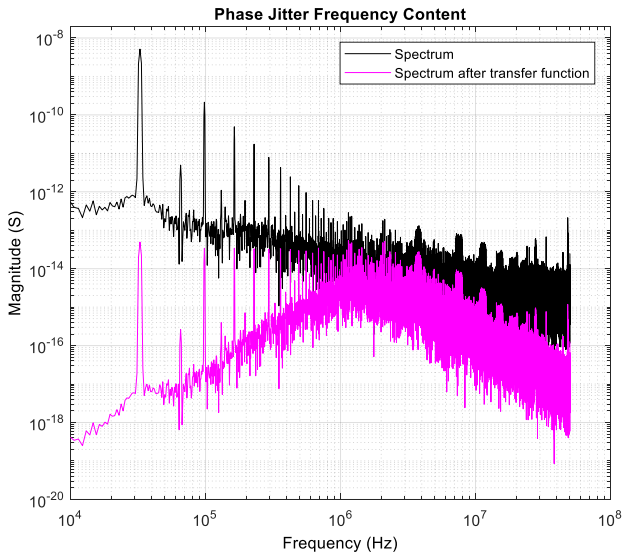

Figure 76: Phase Jitter Frequency Content for X and Y

The resulting eye closure in the time domain is shown in the following, along with the jitter statistics. The beginning and end of the record is truncated. This is needed when the SSC removal algorithm is implemented, as removing spectral spurs is a non-causal, non-linear action that introduces discontinuities at the beginning and end of the inverse transforms.

Figure 77 shows the error between the sampling clock and the data is shown, in time. This is called Y in the transfer functions (y in the time domain). The result RMS value is given, along with the P-P eye closure. In this example approximately 100k periods was used.

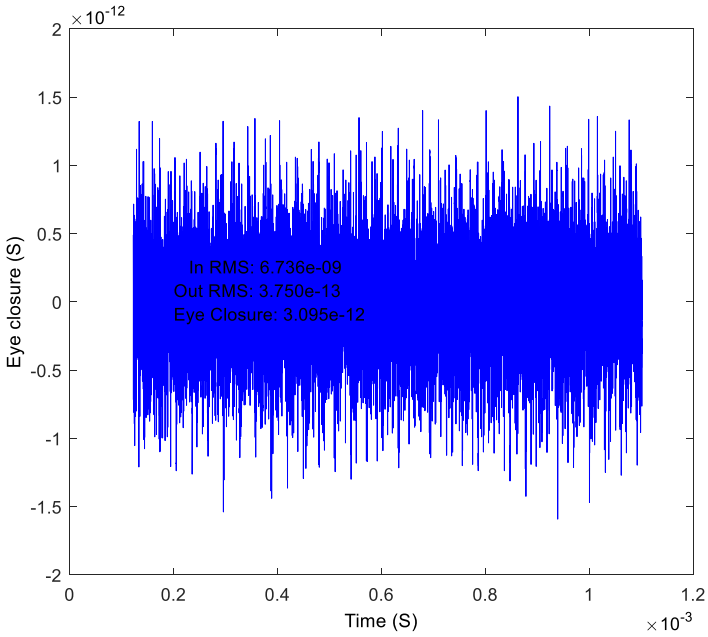

Figure 77: Time Interval Error at Sampling Latch

5.3 Plesochronous System Models

A plesochronous system has two separate clock sources for each component, each with their own frequency and noise profiles. In this case, the CDR operation is vital to properly recover the data. In the case of Ethernet, the CDR is given as a 10 MHz high pass filter. There is no explicit specification in Ethernet for the reference clock, rather the reference clock jitter is contained within the transmitter jitter specification. The optional plesochronous mode in PCI Express, however, maintains a reference clock budget in

addition to a separate transmit budget. This mode is called “separate reference, independent spread” or SRIS mode. As the name implies, spread spectrum is allowed on both devices and the CDR is required to track this very large, low frequency component. Another mode supported is called “separate reference no spread” or SRNS, this is a much simpler mode than SRIS as there is no spread spectrum for the CDR to track. SRNS is very similar to Ethernet, with the exception of a complex third order CDR and a separate reference clock budget.

5.3.1 Ethernet Model

The Ethernet clocking model is one of the simplest, as the reference clock is not budgeted separately in the specification. It is left to the system designer to choose the reference clock and transmit PLL parameters that provide the optimum solution for the jitter transfer. SSC is not allowed, simplifying the problem.

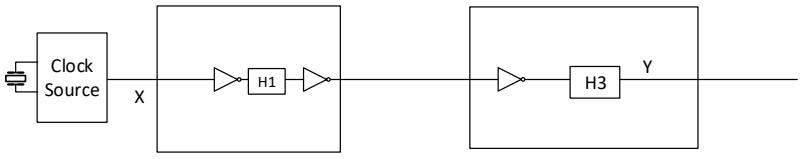

Figure 78: The Ethernet Clocking Model

As shown in Figure 78, the clock source is multiplied to the data rate through a transmit PLL that is not specified, and sent to a CDR that is specified as a 10 MHz first order high-pass filter. The clock source and the transmit PLL is not specified, only the total transmit jitter, measured through H3, is specified. The jitter transfer function is simply

5-11 $$Y = X\ H_1\ H_3.$$

This leaves the component vendor free to choose the requirements of the clock source and the transmit PLL in order to minimize the noise impact. By choosing a low bandwidth transmit PLL and a low jitter clock source, clock jitter impact of less than 20 femtoseconds RMS is typical. The transfer function for a low bandwidth transmitter is shown in Figure 79. Very good jitter rejection to low jitter clock is possible, resulting in minimal impact of the reference clock to the overall timing budget.

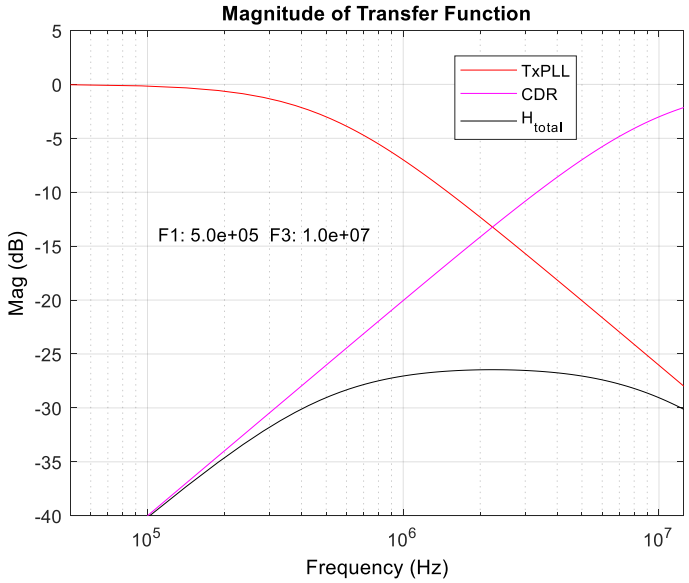

Figure 79: Jitter Transfer Function for Ethernet

5.3.2 PCI Express SRIS Mode

The PCI Express SRIS block diagram is shown in Figure 17. Unlike the plesochronous architecture, the reference clock at the receiver is of no consequence to the jitter transfer function as it carries no information about the source jitter. If the receiver's reference clock is used at all, it is part of the CDR transfer function but is not part of any JTF or specification. Instead of the Ethernet first order 10 MHz high pass, the jitter

transfer function for PCIe is given by a complex third order function that sets the minimum requirements for the CDR design. This function is intended to allow a receiver to fully track SSC to within the budget. The clocking model is identical to the Ethernet model shown in Figure 78, with the new H3 as given in the PCI Express Base Specification. The jitter transfer function for PCI Express, fourth generation, is shown in Figure 80.

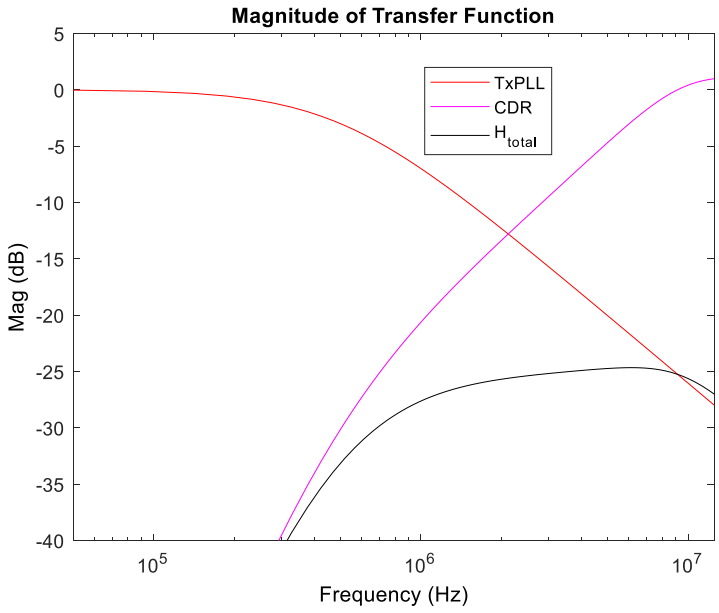

Figure 80: PCI Express SRIS CDR

5.4 Summary

Modeling the total system transfer function requires identifying the significant noise sources in the particular architecture being modeled. The noise sources are quantified, and the PLL transfer functions, if any are evaluated with the worst case design parameters. The appropriate delays are added to the model, and the entire system transfer function is evaluated to get the error, or eye closure, at the receiver. This result can then

be evaluated against the specification limits to check the system for compliance under the assumed noise and transfer function conditions.

This method ensures different devices with differing designs will interoperate reliably under the worst case conditions.

6 Conclusion

Phase noise in the clocking and sampling circuits of a communication system contributes to the probability of a sampling error. The complete model of the phase noise is needed to properly budget a communications link and ensure interoperability between components as intended in the specification.

The refined models of power delivery, CDR tracking and crosstalk noise as presented in this dissertation, have enabled the specification and measurement of sub-picosecond timing calculations for clock jitter. This work has been presented at numerous conferences [3,16-20] and has been incorporated into the products and specifications for the timing budgets of several major interfaces, including PCI Express, USB, and HBM memory interfaces.

The models presented in this work treat the clock jitter as a random Gaussian variable. It is suggested that future clock jitter specifications will need to break the sources of clock jitter into more detailed distributions, such as deterministic or uniform jitter distributions, such as those used to characterize the other components of the TIE analysis, the channel, TX and RX budgets. This will require more advanced statistical method to convolve the jitter into the correct overall distribution, and evaluate the distribution at the appropriate BER. Future work will need to investigate the limits of a Gaussian variable that is bounded due to physical bandwidth limits present in the system.

This work has provided the jitter distribution for binary signaling. Current data rates are at the bandwidth limit of the typical channel, and future investigation will need

to evaluate jitter impact to PAM4 sampling problems. It is likely that these future architectures will require modification to the models presented in order to adapt them to new encoding methods.

References

[1] C. E. Shannon, “A mathematical theory of communication”, The Bell System Technical
Journal, Vol. 27, pp. 379–423, 623–656, July, October, 1948.

[2] D. A. Howe and T. N. Tasset, "Clock jitter estimation based on PM noise measurements," *IEEE International Frequency Control Symposium and PDA Exhibition Jointly with the 17th European Frequency and Time Forum, 2003. Proceedings of the 2003,* 2003, pp. 541-546.

[3] Li, M., Martwick, A., Talbot, G., Wilstrup, J., ” Transfer functions for the reference clock jitter in a serial link: theory and applications”, Test Conference, 2004. Proceedings. International , Oct. 26-28, 2004 Pages:1158 – 1167

[4] J. Montanaro, R. Witek, K. Anne, and A. Black, “A 160-MHz 32-b 0.5-W CMOS RISC microprocessor,” IEEE J. SolidState Circuits, vol. 31, pp. 1703–1714, Nov. 1996.

[5] Best, R. E., “Phase-locked Loops: Design, Simulation, and Applications”, McGraw Hill, 1997

[6] Thomas H. Lee, Kevin S. Donnelly, John T. C. Ho, Jared Zerbe, Mark G. Johnson, and Tom Ishikawa, “A 2.5 V CMOS Delay-Locked Loop for an 18 Mbit, 500 Megabytek DRAM”, IEEE Journal of Solid State Circuits, VOL 29, No. 12, December 1994, pp 1491-1496.

[7] S. Sidiropoulos and M. Horowitz, “A semidigital dual delaylocked loop,” IEEE J. Solid Sfate Circuits, vol. 32, pp. 1683- 1692, Nov. 1997.

[8] W. J. Dally and J. W. Poulton, “Digital Systems Engineering”, Cambridge University Press, 1998.

[9] Poulton, J.W.; Dally, W.J.; Xi Chen; Eyles, J.G.; Greer, T.H.; Tell, S.G.; Gray, C.T., "A 0.54pJ/b 20Gb/s ground-referenced single-ended short-haul serial link in 28nm CMOS for advanced packaging applications," Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2013 IEEE International , vol., no., pp.404,405, 17-21 Feb. 2013

[10]A. X. Widmer, P. A. Franaszek, “A DC-Balanced, Partitioned-Block, 8B/1OB Transmission Code”, IBM J. Res. Develop. Vol. 27, No. 5, September 1983

[11] Ming-ta Hsieh; Sobelman, G.E.; “Clock and data recovery with adaptive loop gain for spread spectrum SerDes applications”, Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium, May 2005 Page(s):4883 - 4886 Vol. 5

[12] S. Sidiropoulos and M. Horowitz, “A semidigital dual delaylocked loop,” IEEE J. Solid Sfate Circuits, vol. 32, pp. 1683- 1692, Nov. 1997.

[13] Ming-ta Hsieh; Sobelman, G.E.; “Clock and data recovery with adaptive loop gain for spread spectrum SerDes applications”, Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium, May 2005 Page(s):4883 - 4886 Vol. 5

[14] X. Wang and A. Martin, "On-die supply-inducecd jitter behavioral modeling," 2013 IEEE 22nd Conference on Electrical Performance of Electronic Packaging and Systems, San Jose, CA, 2013, pp. 147-150.

[15] T. Na and S. Mukhopadhyay, "Behavioral modeling of timing slack variation in digital circuits due to power supply noise," 2016 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, 2016, pp. 281-284.

[16] A. Martwick, “Jitter Modeling, Statistics, and Measurement”, Intel 2017 Clock and Timing Summit, Guadalajara, Jalisco, Mexico, 2017

[17] A. Martwick and J. Drew, "Silicon interposer and TSV signaling," 2015 IEEE 65th Electronic Components and Technology Conference (ECTC), San Diego, CA, 2015, pp. 266-275.

[18] A. Martwick, O. Mikulchenko, et al, “BUDGETING AND MEASURING A HIGH SPEED GRAPHICS MEMORY INTERFACE”, DesignCon 2010, Santa Clara, CA, 2010, 6-WA3, pp. 612-638.

[19] Garmatyuk, D.; Martwick, A., "A Novel Method of Active Power Noise Cancellation in I/O Buffers," Advanced Packaging, IEEE Transactions on , vol.32, no.1, pp.26-34, Feb. 2009

[20] Martwick, A., “Modeling a phase interpolator as a delta-sigma converter”, DesignCon 2008, best paper finalist, Santa Clara, CA, 2008, pp. 908-932.

[21] J.D.H. Alexander, "Clock Recovery from Random Binary Signals", vol. 11, pp. 541 - 542, October 1975 in Electronics Letters. © Institution of Electrical Engineers.